教育部高等学校软件工程专业教学指导委员会规划教材
高 等 学 校 软 件 工 程 专 业 系 列 教 材
中 国 大 学 出 版 社 图 书 奖 优 秀 教 材




实验和项目案例

课件和源码资源

储久良 编著

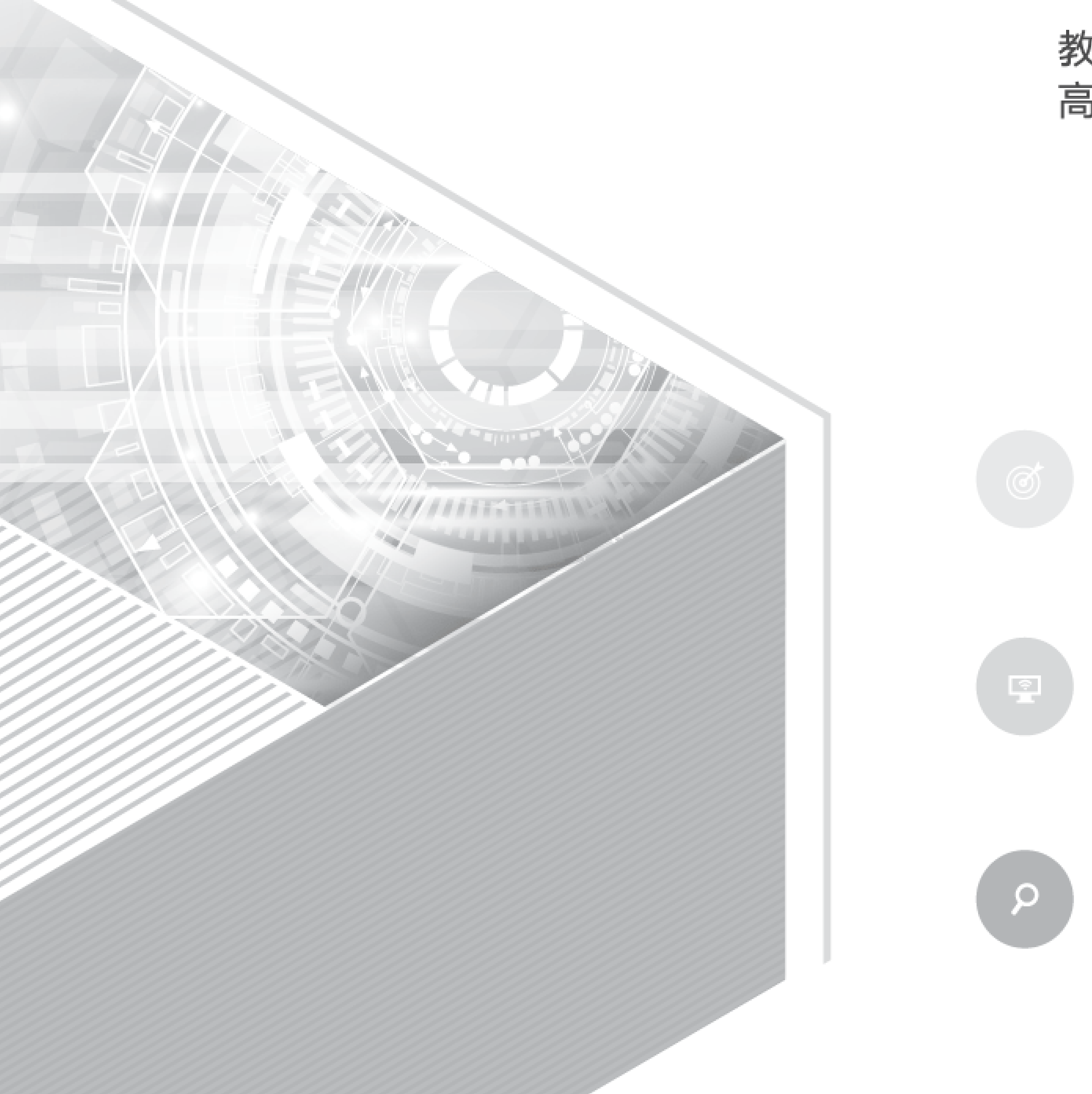
# Web 前端开发技术

## ——HTML5、CSS3、JavaScript (第3版)

微课版
30小时教学视频
全程语音讲解

清華大学出版社

教育部高等学校软件工程专业教学指导委员会规划教材
高 等 学 校 软 件 工 程 专 业 系 列 教 材



◎ 储久良 编著

# Web 前端开发技术

## ——HTML5、CSS3、JavaScript (第3版)

清華大学出版社
北 京

## 内 容 简 介

本书第 3 版紧贴互联网行业发展对 Web 前端开发工程师岗位的新要求，结合众多高校教师的教学反馈意见和建议，在第 2 版的基础上新增加了 HTML5 和 CSS3 相关新特性和新应用。全书详细地介绍了 HTML、CSS、DIV、HTML5 基础和 CSS3 应用、JavaScript、DOM 与 BOM、HTML5 高级应用等部分的基本语法和关键应用。

本书内容编排结构合理，由浅入深，循序渐进地引导读者快速入门，并能提高初级及以上读者的实际应用水平，让读者能够快速适应移动互联网行业对 Web 前端开发工程师岗位的新需求。扫描每章提供的二维码可观看相应知识点的视频讲解。

本书可作为高等学校计算机科学与技术、软件工程、信息管理与信息系统、网络工程、物联网工程、信息科学技术、数字媒体技术、数据科学与技术（大数据管理相关）及其他文、理科相关专业或计算机公共基础的“网页开发与设计”“网站建设与网页制作”“Web 客户端编程”“Web 前端开发技术”“Web 应用技术”等课程教学的教材，也可作为 IT 相关岗位的工程技术人员参考用书，还可以作为初学者的自学读本。



**图书在版编目（CIP）数据**

Web 前端开发技术：HTML5、CSS3、JavaScript/储久良编著. —3 版. —北京：清华大学出版社，2018
（高等学校软件工程专业系列教材）
ISBN 978-7-302-48863-7

Ⅰ. ①W…　Ⅱ. ①储…　Ⅲ. ①超文本标记语言－程序设计②网页制作工具③JAVA 语言－程序设计　Ⅳ. ①TP312.8②TP393.092.2

中国版本图书馆 CIP 数据核字（2017）第 287732 号

**责任编辑：**魏江江　李　晔
**封面设计：**刘　键
**责任校对：**胡伟民
**责任印制：**李红英

**出版发行：**清华大学出版社
**网　　址：**http://www.tup.com.cn, http://www.wqbook.com
**地　　址：**北京清华大学学研大厦 A 座　　**邮　　编：**100084
**社 总 机：**010-62770175　　**邮　　购：**010-62786544
**投稿与读者服务：**010-62776969，c-service@tup.tsinghua.edu.cn
**质量反馈：**010-62772015，zhiliang@tup.tsinghua.edu.cn
**印 装 者：**北京密云胶印厂
**经　　销：**全国新华书店
**开　　本：**185mm×260mm　　**印　　张：**28.75　　**字　　数：**700 千字
**版　　次：**2015 年 11 月第 1 版　2018 年 7 月第 3 版　　**印　　次：**2018 年 7 月第 1 次印刷
**印　　数：**44001～46000
**定　　价：**59.50 元

产品编号：077643-01

# 第3版前言

本书第 2 版自 2016 年 8 月由清华大学出版社出版以来，受到全国各类高等院校的青睐。教材覆盖地域宽广，教材使用层次多样。近年来陆续被武汉大学、重庆大学、吉林大学、北京理工大学、西北农林科技大学、北京邮电大学、东南大学、河海大学、上海大学等 200 多所高等院校选作教材或教学参考书。教材第 1 版获**“第四届中国大学出版社图书奖优秀教材二等奖”**。2016 年第 2 版理论教材与实践教材双双入选**“教育部高等学校软件工程专业教学指导委员会规划教材”**（全国仅 2 部教材入选）。

Web 前端开发技术已经成为 21 世纪高等学校学生及 IT 职员跨入互联网世界的最基础的入门技术。随着“互联网+”模式的不断推广与普及，IT 行业对 Web 前端开发工程师所掌握的知识和能力要求也随之提高了，结合 IT 行业发展的需要和各类高等院校的实际教学反馈，编者在保持前 2 个版本教材原有特色和编写风格的基础上，适时将 HTML5 和 CSS3 等技术补充到教材之中，以期满足当前 IT 行业需要和高等学校培养应用技术型人才的需要。

## 教材编写特色

内容新颖全面。紧贴 Web 前端开发工程师的岗位需求，精心策划教学内容。全面讲解 HTML、CSS、DIV、HTML5 基础和 CSS3 应用、JavaScript、DOM 与 BOM、HTML5 高级应用等内容。

实例真实丰富。从商业网站精选实例，每章再遴选一个经典的综合案例，将本章和相邻章节的知识融会贯通。

讲解图文并茂。使用大量图表、图片进行归纳与分析，以提高教学效率。

代码规范统一。提供风格统一、格式规范的源代码，培养读者良好的编程习惯。

微课视频精美。关键知识和操作技能配套精美的微课视频讲解，让学生无师自通。

## 本次修订内容

第 3 版修订教材共规划了 17 章。保留第 2 版中的第 1～4 章；对第 5 章内容进行更新与重组，将第 2 版中的第 12 章框架中浮动框架部分并入新版的第 5 章中；简化第 6 章图像与多媒体文件；保留第 2 版第 7 章 CSS 基础、第 8 章 DIV 与 SPAN、第 9 章 CSS 样式属性、第 10 章 DIV+CSS 页面布局、第 11 章表格，并将 CSS3 中的属性选择器补充到第 7 章中；删除第 2 版中第 12 章框架；将第 2 版中第 13 章表单改为第 3 版的第 12 章表单；新增第 13 章 HTML5 基础和 CSS3 应用；保留第 2 版中第 14～16 章；删除第 2 版中第 17 章多浏览器兼容性测试、网站调试与发布，更新为新的第 17 章 HTML5 高级应用。同时根据多数使用高校教师的建议和反馈意见对各章节内容和案例进行优化和重组。

限于篇幅，删除第 2 版中每章结尾的“网站赏析”“工具介绍”等课外资源。

## 主要内容

第 1 章和第 2 章重点介绍了 Web 起源、Web 特点与工作原理、Web 前端开发技术、开发工具及 HTML 基础语法和文档结构等知识；第 3～6 章重点介绍了 HTML 网页中格式化文本与段落、列表、超链接与浮动框架、图像与多媒体文件的应用；第 7～10 章重点介绍了 CSS 基础、DIV 与 SPAN、CSS 样式属性、DIV+CSS 页面布局；第 11～12 章重点介绍了表格、表单等页面布局技术；第 13 章重点介绍了 HTML5 基础和 CSS3 应用；第 14～16 章重点介绍了 JavaScript 基础、JavaScript 程序结构、事件分析、DOM 与 BOM 初步应用；第 17 章重点介绍了 HTML5 高级应用。

## 教学资源

为了方便各类高校选用教材进行教学和读者选书自学，第 3 版教材依然提供了大量的实例代码及其他资源。教材中教学案例以统一格式进行命名，如 edu_2_1_1.html 表示第 2 章 2.1 节第 1 个案例。每章资源以子目录形式存放，如 ch5，存放第 5 章的教学资源，有教学案例、图片、音视频等资源。同时还同步改编了配套实验与实践教材《Web 前端开发技术实验与实践——HTML5、CSS3、JavaScript(第 3 版)》，除此之外，我们准备了各种辅助教学材料，包括：

（1）一套完整的教学精简版的 PPT。

（2）一套完整的教学案例代码。

（3）一套完整的教学与实验中所需的图片、文字、音视频素材。

（4）一套完整的练习与实验参考答案。

（5）六套完整的课程考试试卷及参考答案。

（6）提供案例微课视频讲解。

第 3 版修订由储久良负责总体策划、编著、审校。南京理工大学王永利教授、叶庆生副教授，浙江工商大学贾波教授，西北农林科技大学蔚继承副教授，辽宁工程技术大学陈虹副教授、肖振久副教授，常熟理工学院高燕副教授，南华大学赵艳辉副教授，唐山学院党长青教授、顾永军副教授，华北科技学院胡英老师，河南工程学院张劳模副教授，成都大学于曦副教授，泰州学院刘立军副教授、花丽副教授，牡丹江大学谢凤静副教授等对教材的再版工作提出了很多宝贵意见，在此对他们表示感谢。此外，姜枫、袁宝华、曹红根、高广银、李丛、刘立军、花丽、宦臣、沈群、曹诚诚、张晓群、王鑫等教师参与了教材编写工作，对他们的辛苦劳动表示由衷感谢！

本书的修订与再版得到清华大学出版社相关人员的大力支持与合作，在此谨表示衷心感谢。在修订过程中，编者参阅了大量的 Web 前端开发、JavaScript 应用、HTML5 和 CSS3 相关等方面的书籍与网络资料，在此对这些书籍与资料的作者表示感谢。由于移动互联网技术发展迅速，加上编者水平有限，书中的不足在所难免，恳请各位专家和读者批评指正。

编　者

2017 年 07 月于苏州方圆·云山诗意

# 目　录

**第 1 章　Web 前端开发技术综述** ······ 1
1.1　Web 概述 ······ 1
1.1.1　Web 的起源 ······ 2
1.1.2　Web 的特点 ······ 3
1.1.3　Web 工作原理 ······ 3
1.1.4　Web 相关概念 ······ 4
1.2　Web 前端开发工程师的职业需求 ······ 6
1.2.1　Web 前端开发的由来 ······ 6
1.2.2　Web 前端开发工程师的职业要求 ······ 6
1.3　Web 前端开发技术 ······ 7
1.3.1　HTML ······ 7
1.3.2　CSS ······ 8
1.3.3　JavaScript ······ 9
1.3.4　HTML DOM ······ 9
1.3.5　BOM ······ 10
1.3.6　AJAX ······ 10
1.3.7　jQuery ······ 11
1.4　Web 前端开发工具 ······ 11
1.4.1　EditPlus ······ 11
1.4.2　Adobe Dreamweaver ······ 11
1.4.3　Sublime Text ······ 12
1.4.4　WebStorm ······ 12
1.4.5　HBuilder ······ 13
1.5　浏览器工具 ······ 13
1.5.1　Internet Explorer ······ 14
1.5.2　Google Chrome ······ 14
1.5.3　Mozilla Firefox ······ 14
1.5.4　Safari ······ 14
1.5.5　Opera ······ 14
1.6　综合实例 ······ 15
本章小结 ······ 16

练习与实验……16
练习 1……16
实验 1……17

第 2 章 HTML 基础……18

2.1 HTML 文档结构……18
2.2 头部 head……19
2.2.1 标题 title 标记……19
2.2.2 元信息 meta 标记……20
2.3 主体 body……22
2.3.1 body 标记……22
2.3.2 body 标记属性……23
2.4 HTML 基本语法……25
2.4.1 标记类型……25
2.4.2 HTML 属性……26
2.5 注释……27
2.6 HTML 文档编写规范……28
2.6.1 HTML 代码书写规范……28
2.6.2 HTML 文档命名规则……29
2.7 HTML 文档类型……30
2.7.1 <!DOCTYPE>标记……30
2.7.2 DTD 类型……30
2.8 综合实例……31
本章小结……32
练习与实验……32
练习 2……32
实验 2……33

第 3 章 格式化文本与段落……34

3.1 Web 页面初步设计……34
3.1.1 向 Web 页面添加文字信息……34
3.1.2 标题字标记……35
3.1.3 添加空格与特殊符号……36
3.2 格式化文本标记……37
3.2.1 文本修饰标记……37
3.2.2 计算机输出标记……38
3.2.3 引用和术语标记……38
3.2.4 字体 font 标记……39
3.3 段落与排版标记……40

3.3.1 段落 p 标记……40
3.3.2 换行 br 标记……41
3.3.3 水平分隔线 hr 标记……41
3.3.4 拼音/音标注释 ruby 标记和 rt/rp 标记……42
3.3.5 段落缩进 blockquote 标记……43
3.3.6 预格式化 pre 标记……44
3.4 综合实例……45
本章小结……46
练习与实验……46
练习 3……46
实验 3……47

**第 4 章 列表**……49

4.1 列表概述……49
4.2 无序列表……49
4.3 有序列表……51
4.4 列表嵌套……53
4.5 定义列表……54
4.6 综合实例……56
本章小结……57
练习与实验……57
练习 4……57
实验 4……58

**第 5 章 超链接与浮动框架**……59

5.1 超链接概述……59
5.2 超链接语法、路径及分类……59
5.2.1 超链接语法……59
5.2.2 超链接路径……61
5.2.3 超链接分类……62
5.3 超链接的应用……62
5.3.1 创建 HTTP 文件下载超链接……62
5.3.2 创建 FTP 站点访问超链接……62
5.3.3 创建图像超链接……63
5.3.4 创建电子邮件超链接……63
5.3.5 创建页面书签链接……65
5.4 浮动框架……67
5.5 综合实例……69
本章小结……70

练习与实验……70
练习 5……70
实验 5……71

**第 6 章　图像与多媒体文件**……73

6.1　图像……73
6.1.1　插入图像……73
6.1.2　设置图像的替代文本……75
6.1.3　设置图像的高度和宽度……75
6.1.4　设置图像的边框……76
6.1.5　设置图像对齐方式……77
6.1.6　设置图像的间距……77
6.1.7　设置图像热区链接……77
6.2　滚动文字……79
6.2.1　添加滚动文字……79
6.2.2　设置滚动文字背景颜色与滚动循环……80
6.2.3　设置滚动方向与滚动方式……80
6.2.4　设置滚动速度与滚动延迟……81
6.2.5　设置滚动范围与滚动空白空间……81
6.3　音频、视频及 Flash 文件……82
6.4　综合实例……84
本章小结……85
练习与实验……85
练习 6……85
实验 6……86

**第 7 章　CSS 基础**……88

7.1　CSS 概念……88
7.1.1　CSS 的基本概念……88
7.1.2　传统 HTML 的缺点……88
7.1.3　CSS 的特点……89
7.1.4　CSS 的优势……89
7.1.5　CSS 的编辑方法……89
7.2　使用 CSS 控制 Web 页面……90
7.2.1　CSS 基本语法……90
7.2.2　CSS 选择器类型……91
7.2.3　CSS 选择器声明……95
7.2.4　CSS 定义与引用……96
7.3　CSS 继承与层叠……101

7.4　综合实例 102
本章小结 105
练习与实验 106
练习 7 106
实验 7 106

**第 8 章　DIV 与 SPAN** 108

8.1　DIV 图层 108
8.1.1　DIV 定义 108
8.1.2　DIV 应用 109
8.2　图层嵌套与层叠 110
8.2.1　DIV 嵌套 110
8.2.2　DIV 层叠 111
8.3　div 标记与 span 标记 112
8.4　综合实例 114
本章小结 118
练习与实验 118
练习 8 118
实验 8 119

**第 9 章　CSS 样式属性** 120

9.1　CSS 属性值中的单位 120
9.1.1　绝对单位 120
9.1.2　相对单位 120
9.2　CSS 字体样式 121
9.2.1　字体大小 font-size 属性 121
9.2.2　字体样式 font-style 属性 122
9.2.3　字体系列 font-family 属性 122
9.2.4　字体变体 font-variant 属性 123
9.2.5　字体粗细 font-weight 属性 124
9.2.6　字体 font 属性 124
9.3　CSS 文本样式 125
9.3.1　字符间距 letter-spacing 属性 125
9.3.2　行距 line-height 属性 125
9.3.3　首行缩进 text-indent 属性 126
9.3.4　字符装饰 text-decoration 属性 127
9.3.5　英文大小写转换 text-transform 属性 127
9.3.6　水平对齐 text-align 属性 128
9.3.7　垂直对齐 vertical-align 属性 128

9.4 CSS 颜色与背景 ······ 130
9.4.1 颜色 color 属性 ······ 130
9.4.2 背景 background 属性 ······ 131
9.5 CSS 列表样式 ······ 134
9.6 CSS 盒模型 ······ 136
9.6.1 CSS 盒模型结构 ······ 136
9.6.2 边界属性设置 ······ 137
9.6.3 边框属性设置 ······ 138
9.6.4 填充属性设置 ······ 141
9.7 综合实例 ······ 142
本章小结 ······ 146
练习与实验 ······ 146
练习 9 ······ 146
实验 9 ······ 147

**第 10 章 DIV+CSS 页面布局** ······ 149

10.1 页面布局设计 ······ 149
10.1.1 “三行模式”或“三列模式” ······ 149
10.1.2 “三行二列”“三行三列”模式 ······ 150
10.1.3 多行多列复杂模式 ······ 152
10.2 导航菜单设计 ······ 154
10.2.1 一级水平导航菜单 ······ 154
10.2.2 二级水平导航菜单 ······ 156
10.3 综合实例 ······ 162
本章小结 ······ 168
练习与实验 ······ 168
练习 10 ······ 168
实验 10 ······ 169

**第 11 章 表格** ······ 170

11.1 表格概述 ······ 170
11.2 表格标记 ······ 171
11.3 表格属性设置 ······ 173
11.3.1 表格边框属性 ······ 174
11.3.2 表格的宽度和高度属性 ······ 174
11.3.3 表格背景颜色与背景图像属性 ······ 174
11.3.4 表格边框样式属性 ······ 176
11.3.5 表格单元格间距、单元格边距属性 ······ 177
11.3.6 表格水平对齐属性 ······ 178

11.4 设置表格行的属性 180
11.5 设置单元格的属性 181
11.5.1 表格单元格跨行属性 182
11.5.2 表格单元格跨列属性 182
11.6 表格嵌套 183
11.7 综合实例 185
本章小结 189
练习与实验 189
练习 11 189
实验 11 190

**第 12 章 表单** 191

12.1 表单概述 191
12.2 定义域和域标题 193
12.3 表单信息输入 194
12.3.1 单行文本输入框 194
12.3.2 密码输入框 195
12.3.3 复选框 196
12.3.4 单选按钮 196
12.3.5 图像按钮 198
12.3.6 提交按钮 199
12.3.7 重置按钮 199
12.3.8 普通按钮 200
12.3.9 文件选择框 201
12.3.10 隐藏框 201
12.4 多行文本输入框 203
12.5 下拉列表框 204
12.6 综合实例 205
本章小结 208
练习与实验 208
练习 12 208
实验 12 209

**第 13 章 HTML5 基础与 CSS3 应用** 210

13.1 HTML5 概述 210
13.1.1 HTML5 的八个特性 211
13.1.2 HTML5 的优势 212
13.1.3 HTML5 新增结构元素及页面元素 213
13.1.4 HTML5 废除的元素与属性 214

13.1.5　浏览器支持与选择 ······ 215
13.2　HTML5 文档结构 ······ 216
13.2.1　HTML5 页面结构 ······ 216
13.2.2　HTML5 新增结构元素 ······ 217
13.3　HTML5 新增页面元素 ······ 221
13.3.1　hgroup 标记 ······ 222
13.3.2　figure 标记与 figcaption 标记 ······ 222
13.3.3　mark 标记与 time 标记 ······ 223
13.3.4　details 标记与 summary 标记 ······ 224
13.3.5　progress 标记与 meter 标记 ······ 225
13.3.6　input 标记与 datalist 标记 ······ 226
13.4　HTML5 表单 ······ 227
13.4.1　HTML5 新增的表单属性 ······ 227
13.4.2　HTML5 新增的表单元素 ······ 231
13.4.3　HTML5 新增的 input 类型 ······ 232
13.5　HTML5 视频与音频 ······ 236
13.5.1　video 标记及属性 ······ 236
13.5.2　audio 标记及属性 ······ 238
13.6　CSS3 基础应用 ······ 239
13.6.1　CSS3 新特性 ······ 239
13.6.2　CSS3 浏览器兼容性 ······ 239
13.6.3　CSS3 边框 ······ 240
13.6.4　CSS3 转换 transform 属性 ······ 247
13.6.5　CSS3 过渡 transition 属性 ······ 251
13.6.6　CSS3 动画 animation ······ 253
13.6.7　CSS3 多列属性 ······ 256
13.6.8　CSS3 文本效果 ······ 258
13.7　综合实例 ······ 260
本章小结 ······ 262
练习与实验 ······ 262
练习 13 ······ 262
实验 13 ······ 263

**第 14 章　JavaScript 基础** ······ 265

14.1　JavaScript 概述 ······ 265
14.1.1　JavaScript 简介 ······ 265
14.1.2　第一个 JavaScript 程序 ······ 266
14.1.3　JavaScript 放置的位置 ······ 267
14.2　JavaScript 程序 ······ 270

14.2.1 语句和语句块 ······ 270
14.2.2 代码 ······ 271
14.2.3 消息对话框 ······ 271
14.2.4 JavaScript 注释 ······ 274
14.3 标识符和变量 ······ 274
14.3.1 命名规范 ······ 274
14.3.2 数据类型 ······ 275
14.3.3 变量 ······ 278
14.3.4 转义字符 ······ 278
14.4 运算符和表达式 ······ 279
14.4.1 算术运算符和表达式 ······ 279
14.4.2 关系运算符和表达式 ······ 281
14.4.3 逻辑运算符和表达式 ······ 283
14.4.4 赋值运算符和表达式 ······ 284
14.4.5 位运算符和表达式 ······ 284
14.4.6 条件运算符和表达式 ······ 286
14.4.7 其他运算符和表达式 ······ 286
14.5 JavaScript 程序控制结构 ······ 287
14.5.1 顺序结构 ······ 287
14.5.2 分支结构 ······ 288
14.5.3 循环结构 ······ 293
14.6 JavaScript 函数 ······ 301
14.6.1 常用系统函数 ······ 301
14.6.2 自定义函数 ······ 310
14.6.3 带参数返回的 return 语句 ······ 311
14.6.4 函数变量的作用域 ······ 312
14.7 综合实例 ······ 313
本章小结 ······ 317
练习与实验 ······ 317
练习 14 ······ 317
实验 14 ······ 318

**第 15 章 JavaScript 事件分析 ······ 319**

15.1 JavaScript 事件概述 ······ 319
15.1.1 事件类型 ······ 319
15.1.2 事件句柄 ······ 320
15.1.3 事件处理 ······ 321
15.1.4 事件处理程序的返回值 ······ 324
15.2 表单事件 ······ 326

15.2.1 获得焦点与失去焦点事件 ······ 326
15.2.2 提交及重置事件 ······ 327
15.2.3 改变及选择事件 ······ 328
15.3 鼠标事件 ······ 329
15.3.1 鼠标单、双击事件 ······ 330
15.3.2 鼠标移动事件 ······ 331
15.4 键盘事件 ······ 332
15.5 窗口事件 ······ 333
15.6 综合实例 ······ 334
本章小结 ······ 336
练习与实验 ······ 336
练习 15 ······ 336
实验 15 ······ 337

**第 16 章 DOM 和 BOM ······ 339**

16.1 JavaScript 常用对象 ······ 339
16.1.1 Array ······ 340
16.1.2 Date ······ 342
16.1.3 Math ······ 345
16.1.4 Number ······ 347
16.1.5 String ······ 347
16.1.6 Boolean ······ 350
16.2 HTML DOM ······ 351
16.2.1 DOM 简介 ······ 351
16.2.2 DOM 节点树 ······ 351
16.2.3 DOM 节点 ······ 352
16.2.4 DOM 节点访问 ······ 353
16.2.5 DOM 节点操作 ······ 357
16.3 BOM ······ 363
16.3.1 window 对象 ······ 363
16.3.2 Navigator 对象 ······ 366
16.3.3 Screen 对象 ······ 367
16.3.4 History 对象 ······ 368
16.3.5 Location 对象 ······ 369
16.4 综合实例 ······ 370
本章小结 ······ 380
练习与实验 ······ 380
练习 16 ······ 380
实验 16 ······ 381

第 17 章　HTML5 高级应用 ········ 383

17.1　HTML5 Web Storage ········ 383
17.1.1　localStorage 对象 ········ 383
17.1.2　sessionStorage 对象 ········ 384
17.1.3　浏览器端数据库 IndexedDB ········ 386
17.2　HTML5 Canvas 画布 ········ 396
17.2.1　Canvas 标记 ········ 396
17.2.2　Canvas 坐标 ········ 398
17.2.3　Canvas 路径 ········ 399
17.2.4　Canvas 绘制线段 ········ 400
17.2.5　Canvas 绘制文本 ········ 402
17.2.6　Canvas 渐变 ········ 402
17.2.7　Canvas 绘制图像 ········ 403
17.3　HTML5 拖放 ········ 406
17.3.1　设置元素为可拖放 ········ 406
17.3.2　拖放事件 ········ 407
17.3.3　dataTransfer 对象 ········ 407
17.3.4　拖放操作实现步骤 ········ 408
17.4　HTML5 Web Worker ········ 410
17.4.1　Web Worker 的工作原理 ········ 410
17.4.2　创建 Web Worker 文件 ········ 410
17.4.3　创建 Web Worker 对象 ········ 410
17.4.4　终止 Web Worker ········ 411
17.5　综合实例 ········ 412
本章小结 ········ 419
练习与实验 ········ 420
练习 17 ········ 420
实验 17 ········ 421

附录 A　模拟试卷 1（120 分） ········ 423

附录 B　模拟试卷 2（120 分） ········ 431

参考文献 ········ 440

# 第 1 章　Web 前端开发技术综述

本章学习目标

Web 是一种典型的分布式应用结构。Web 应用中的信息交换与传输都要涉及客户端和服务器端。因此，Web 开发技术分为客户端开发技术（又名“Web 前端开发技术”）和服务器端开发技术两大类。Web 前端（客户端）的主要任务是信息内容的呈现和用户界面（User Interface，UI）设计。Web 前端开发技术主要包括 HTML、CSS、JavaScript、DOM、BOM、AJAX、jQuery 及其他插件技术。学习完本章后读者对 Web 前端开发技术能有一个总体的认识。

Web 前端开发工程师应掌握以下内容：

- 了解 Web 发展历史。
- 了解 Web 前端开发工程师的职业需求。
- 掌握 Web 网站相关的基本概念。
- 理解各种 Web 前端开发技术及其在 Web 网页中的作用。
- 熟悉各种常用的 Web 前端开发工具、浏览器工具，并学会使用主流开发工具。

## 1.1　Web 概述

1980 年 Tim Berners-Lee（蒂姆·伯纳斯·李）在欧洲核子研究组织(European Organisation for Nuclear Research，CERN）中最大的欧洲核子物理实验室（European Particle Physics Laboratory，EPPL）工作时建议建立一个以超文本系统为基础的项目，能使得科学家之间分享和更新他们的研究结果。他与 Robert Cailliau（罗伯特·卡里奥）一起建立了一个叫作 ENQUIRE 的原型系统。

1984 年 Tim Berners-Lee 重返欧洲核子物理实验室，他恢复了自己过去的工作，并创造了万维网。为此他写了世界上第一个客户端浏览器（World Wide Web，也是一个编辑器）和第一个 Web 服务器 httpd（超文本传输协议守护进程）。Tim Berners-Lee 建立了世界上的第一个网站，网址是 http://info.cern.ch/hypertext/WWW/TheProject.html，现在的网址是 http://info.cern.ch/，如图 1-1 所示，并于 1991 年 8 月 6 日发布。它解释了什么是万维网，如何使用网页浏览器和如何建立一个 Web 服务器等。Tim Berners-Lee 后来在这个网站里列举了其他网站，因此它也是世界上第一个万维网导航站点。



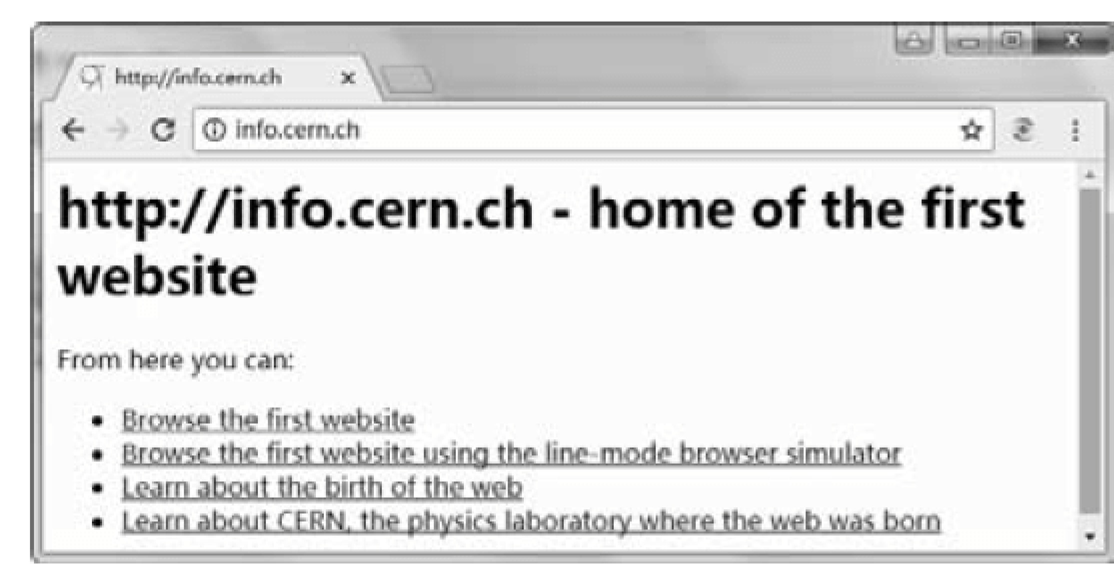



图 1-1　万维网发明人和世界上的第一个网站

## 1.1.1　Web 的起源

最早的网络构想可以追溯到 1980 年 Tim Berners-Lee 构建的 ENQUIRE 项目。这是一个类似于维基百科(wiki)的超文本在线编辑数据库。尽管这与现在使用的万维网大不相同，但是它们有许多相同的核心思想，甚至还包括一些 Tim Berners-Lee 的万维网之后的下一个项目语义网中的构想。

1989 年 3 月，Tim Berners-Lee 撰写了 Information Management: A Proposal（关于信息化管理的建议）一文，文中提及 ENQUIRE 并且描述了一个更加精巧的管理模型。1990 年 11 月 12 日他和 Robert Cailliau 合作提出了一个更加正式的关于万维网的建议。在 1990 年 11 月 13 日他在一台 NeXT 工作站（正式名称是 NeXT Computer）上写了第一个网页以实现他文中的想法，NeXT 工作站如图 1-2 所示，后来这台工作站成为世界上第一台互联网服务器。




图 1-2　Tim Berners-Lee 使用的 NeXT 工作站

在那年的圣诞假期，Tim Berners-Lee 设计了一套开展网络工作所必需的所有工具：第一个万维网浏览器（同时也是编辑器）和第一个 Web 服务器。

1991 年 8 月 6 日，他在 alt.hypertext 新闻组上发布了万维网项目简介的文章，这一天标志着因特网上万维网公共服务的首次亮相。

万维网中至关重要的概念——超文本起源于 20 世纪 60 年代的几个项目。例如 Ted Nelson（泰德·尼尔森）的 Project Xanadu 项目和 Douglas Engelbart（道格拉斯·英格巴特）的 NLS 项目。这两个项目的灵感都是来源于 Vannevar Bush（万尼瓦尔·布什）在其 1945

年的论文《和我们想的一样》中为微缩胶片设计的“记忆延伸”（memex）系统。

Tim Berners-Lee 的另一个重大突破是将超文本嫁接到因特网上。在他的书 *Weaving the Web*（《编织网络》）中，他解释说他曾一再向这两种技术的使用者们建议它们的结合是可行的，但是却没有任何人响应他的建议，他最后只好自己执行了这个计划。他发明了一个全球网络资源唯一认证的系统：统一资源标识符（Uniform Resource Identifier，URI）。

为了让 World Wide Web 不被少数人所控制，Tim 组织成立了 World Wide Web Consortium，即通常所说的 W3C，致力于“引导 Web 发挥其最大潜力”。我们所熟知的 HTML 协议各个版本，都出自 W3C 会议。可贵的是，W3C 的 HTML 规范是以“建议”的形式发布，并不强迫任何厂商或个人接受。至于微软利用 HTML 协议的开放性扩展自己的标准，打败 Netscape，应该是 Tim 始料未及的事件。

## 1.1.2　Web 的特点

**1．易导航和图形化的界面**

Web 非常流行的一个很重要的原因就在于它可以在一页上同时显示色彩丰富的图形和文本，而在 Web 之前因特网上的信息只有文本形式。Web 具有可以将图形、音频、视频等信息集于一体的特性。同时，Web 导航非常方便，只需要从一个链接跳到另一个链接，就可以在各个页面、各个站点之间进行浏览了。

**2．与平台无关性**

无论计算机系统是什么平台，都可以通过因特网访问 WWW。浏览 WWW 对计算机系统平台没有任何限制。从 Windows、UNIX、Macintosh 以及其他平台都能通过一种叫作浏览器（Browser）的软件实现对 WWW 的访问，如 Chrome、IE、Firefox 等。

**3．分布式结构**

大量的图形、音频和视频信息会占用相当大的磁盘空间，事先很难预知信息的多少。对于 Web 来说，信息可以放在不同的站点上，而没有必要集中在一起，浏览时只需要在浏览器中指明这个站点就可以了。这样就使物理上不一定在一个站点的信息在逻辑上是在一体的，从用户的角度来看这些信息也是一体的。

**4．动态性**

由于各 Web 站点的信息包含站点本身的信息，信息的提供者可以经常对站上的信息进行更新与维护。一般来说，各信息站点都尽量保证信息的时效性，所以 Web 站点上的信息需要动态更新，这一点可以通过信息的提供者实时维护。

**5．交互性**

Web 的交互性首先表现在它的超链接上，用户的浏览顺序和所访问的站点完全由用户自己决定。另外通过表单 Form 的形式可以从服务器方获得动态的信息。用户通过填写 Form 可以向服务器提交请求，服务器根据用户的请求返回响应信息。

## 1.1.3　Web 工作原理

用户通过客户端浏览器访问因特网上的网站或者其他网络资源时，通常需要在客户端的浏览器的地址栏中键入需要访问网站的统一资源定位符（Uniform Resource Locator，URL），或者通过超链接方式链接到相关网页或网络资源；然后通过域名服务器进行全球域

名解析，并根据解析结果决定访问指定 IP 地址（IP address）的网站或网页。

获取网站的 IP 后，客户端的浏览器向指定 IP 地址上的 Web 服务器发送一个 HTTP（Hypertext Transfer Protocol，超文本传输协议）请求；在通常情况下，Web 服务器会很快响应客户端的请求，将用户所需要的 HTML 文本、图片和构成该网页的其他一切文件发送回用户。如果需要访问数据库系统中的数据时，Web 服务器会将控制权转给应用服务器，根据 Web 服务器的数据请求读写数据库，并进行相关数据库的访问操作，应用服务器将数据查询响应发送给 Web 服务器，由 Web 服务器再将查询结果转发给客户端的浏览器；浏览器将客户端请求的页面内容组成一个网页显示给用户。这就是 Web 的工作原理，如图 1-3 所示。

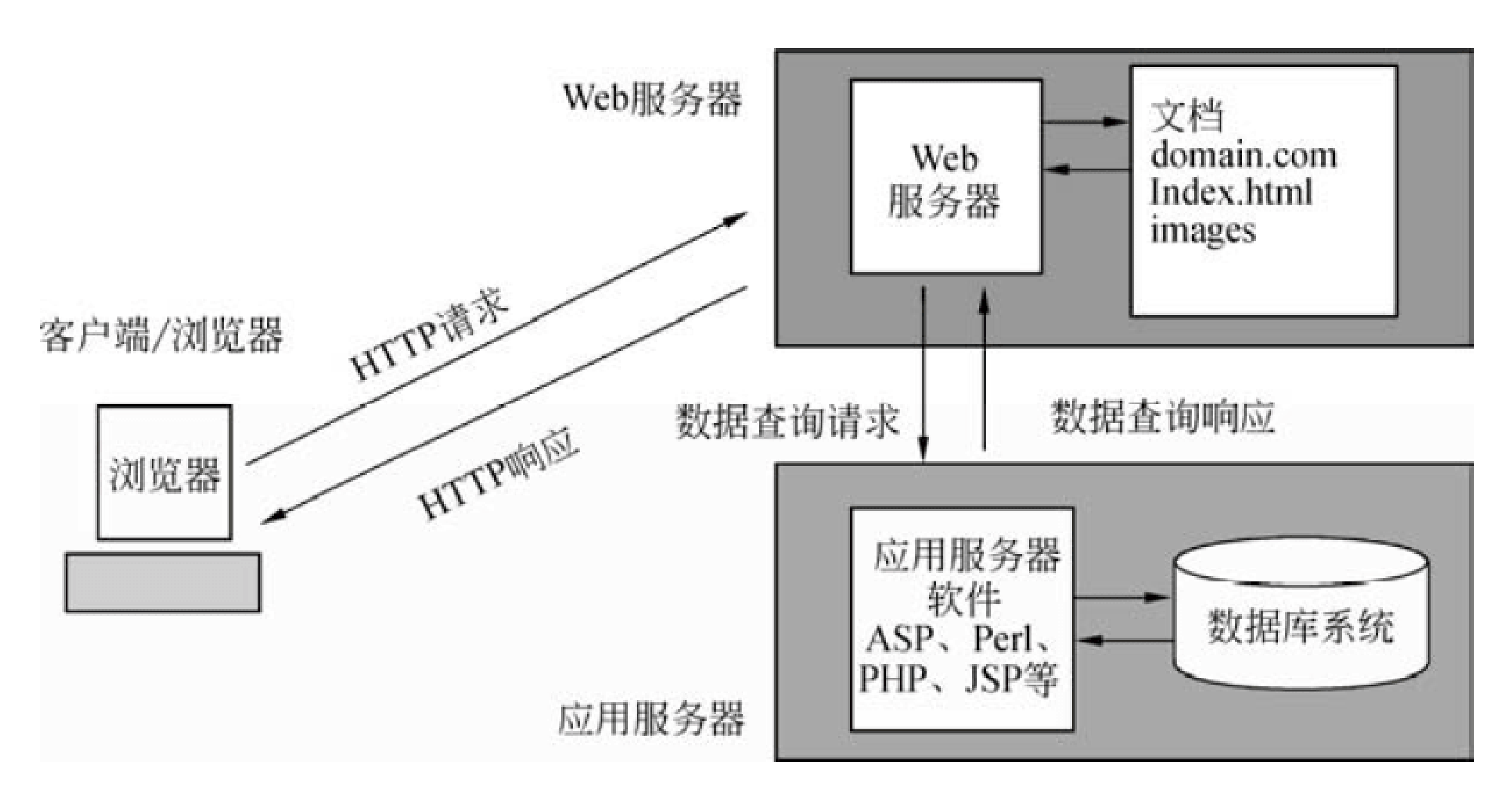



图 1-3 Web 工作原理

大多数网站的网页中包含很多超链接，有内链接和外链接。通过超链接可以设置资源下载、页面浏览及链接其他网络资源。像这样通过超链接，把有用的相关资源组织在一起的集合，就形成了一个所谓的信息的“网”。这个网运行在因特网上，使用十分方便，就构成了最早在 1990 年初 Tim Berners-Lee 所说的万维网。

## 1.1.4 Web 相关概念

### 1. URL 统一资源定位器

URL（Uniform Resource Locator）即统一资源定位器（或统一资源定位符），可以理解为网页地址。如同在网络上的门牌，是因特网上标准的资源的地址（Address）。由 Tim Berners-Lee 发明用来作为万维网的地址。现在它已经被万维网联盟编制为因特网标准 RFC1738。

URL 由协议、主机域名及路径和文件名三个部分组成，其构成如下所示：

```
协议类型://服务器地址（端口号）/路径/文件名
```

第一部分是协议（或称为服务类型），如表 1-1 所示；

第二部分是资源主机的域名或 IP 地址（包括端口号），http 默认的端口号是 80；

第三部分是主机资源的具体地址，如目录和文件名等。

第一部分和第二部分之间用“://”符号隔开，第二部分和第三部分用“/”符号隔开。第一部分和第二部分是不可缺少的，第三部分有时可以省略。下面是一些例子：

```
http://www.edu.cn/kexuetansuo_12385/index.shtml
ftp://ftp.pku.edu.cn/
http://58.195.195.22:8089/web/index.html
```

**表 1-1　URL 中的协议类型**

| 序　　号 | 服务（协议）类型 | 含　　义 |
|---|---|---|
| 1 | http | 超文本传输协议 |
| 2 | https | 用加密传送的超文本传输协议 |
| 3 | ftp | 文件传输协议 |
| 4 | mailto | 电子邮件地址 |
| 5 | ldap | 轻量目录访问协议 |
| 6 | news | Usenet 新闻组 |
| 7 | file | 本地计算机或网上分享的文件 |
| 8 | gopher | Internet Gopher protocol（因特网查找协议） |

**2．Web 服务器**

Web 服务器也称为网站，是指在因特网上提供 Web 访问服务的站点，是由计算机软件和硬件组成的有机整体。网站一般采用 PHP、JSP、ASP 等技术开发而成 B/S（Browser/Server）架构，一般由若干个网页有序地组织在一起，第一个网页也称为主页，所以主页的设计非常重要。通常需要为 Web 服务器配置 IP 地址和域名，才能对外提供 Web 服务。

**3．超链接**

Web 页面一般是由若干超链接构成。所谓超链接（Hyper Link），是指从一个网页指向另一个目标的连接关系，这个目标可以是另一个网页，也可以是相同网页上的不同位置，还可以是一个图片、一个电子邮件地址、一个文件，甚至是一个应用程序。

文本超链接在浏览器中表现为带有下画线的文字，将鼠标移动到文字上时，浏览器会将光标转变为手的形状。如图 1-4 所示为世界上第一个网页上的超链接。

网页中超链接的格式如下所示：

```
<a  href="http://info.cern.ch/hypertext/WWW/TheProject.html">Browse  the
first website</a>
```




图 1-4　世界上的第一个网站

# 1.2　Web 前端开发工程师的职业需求

2005 年以后，互联网进入 Web 2.0 时代，网站的客户端（前端）由此发生了翻天覆地的变化。网站不再是 Web 1.0 时代承载单一的文字和图片的信息提供者，富媒体（Rich Media，RM）让网站的内容更加生动，软件化的交互形式为用户提供了更好的使用体验。Web 2.0 时代更注重用户的交互作用，用户成为网站内容的浏览者和提供者，网站需要前端技术来实现。

## 1.2.1　Web 前端开发的由来

Web 前端开发是从网页制作演变而来的，名称上有很明显的时代特征。Web 前端开发工程师是一个很新的职业，在国内乃至国际上真正开始受到重视的时间不超过 10 年。

随着 Web 2.0 概念的普及和 W3C 组织的推广，网站重构的影响力正以惊人的速度增长。HTML+CSS 布局、DHTML 和 AJAX 像一阵旋风，铺天盖地席卷而来，包括新浪、搜狐、网易、腾讯、淘宝等在内的各行各业的 IT 企业都对自己的网站进行了重构。

随着人们对用户体验的要求越来越高，前端开发的技术难度越来越大，Web 前端开发工程师这一职业终于从设计和制作不分的局面中独立出来。

我国互联网行业的发展呈现迅猛的增长势头，对网站开发、设计制作的人才需求随之大增。Web 前端开发正是采用 HTML、CSS、div、JavaScript、DOM、AJAX 等技术实现网站整体风格优化与改善用户体验的工作。在欧美技术发达国家里，前端开发和后台开发人员的比例为 1∶1，而在我国目前依旧在 1∶3 以下，人才缺口较大。截至 2016 年 6 月，中国网站数量为 454 万个。目前我国各行业领域几乎都要建设自己的网站，网络调查结果表明，未来几年，国内各大行业对 Web 前端开发方面的人才需求量将会大幅度提升，Web 前端开发工程师也会日益受到重视。

## 1.2.2　Web 前端开发工程师的职业要求

Web 前端开发工程师的职业要求是利用 HTML、CSS、JavaScript、DOM、AJAX 等各种 Web 技术进行产品的界面开发。编写标准、优化的代码，并增加交互动态功能，开发 JavaScript 以及 Flash 模块，同时结合后台开发技术模拟整体效果，进行富互联网应用（Rich Internet Applications，RIA）的 Web 开发，致力于通过技术改善用户体验，这需要对用户体验、交互操作流程及用户需求有深入理解。

一位优秀的 Web 前端开发工程师在知识体系上既要有广度，又要有深度。以前会 Photoshop 和 Dreamweaver 就可以制作网页，现在只掌握这些已经远远不够了。无论是在开发难度上，还是在开发方式上，现在的网页制作都更接近传统的网站后台开发，所以现在不再叫网页制作，而是叫 Web 前端开发。Web 前端开发在产品开发环节中的作用变得越来越重要，需要更专业的前端工程师才能做好，这方面的专业人才近年来备受青睐。Web 前端开发是一项很特殊的工作，涵盖的知识面非常广，要求既有具体的技术，又有抽象的理念。简单地说，其主要职责就是把网站的界面更好地呈现给用户。

Web 前端开发工程师具体技术要求如下：

（1）必须掌握基本的 Web 前端开发技术，其中包括 HTML5、CSS3、JavaScript、DOM、BOM、Ajax 等。在掌握这些技术的同时，还要清楚地了解它们在不同浏览器上的兼容性情况、渲染原理和存在的问题。

（2）必须掌握网站性能优化、搜索引擎优化（SEO）和服务器端开发技术的基础知识。

（3）必须学会运用各种 Web 前端开发与测试工具进行辅助开发。

（4）除了要掌握技术层面的知识，还要掌握理论层面的知识，包括 Web 视觉设计、网站配色、网站交互设计模式、代码的可维护性、组件的易用性、分层语义模板和浏览器分级支持等。

# 1.3　Web 前端开发技术

随着因特网技术飞速发展与普及，Web 技术也在同步发展，并且应用领域越来越宽广。WWW（World Wide Web）已经是这个时代不可或缺的信息传播载体。全球范围内的资源互通互访、开放共享已经成为 WWW 最有实际应用价值的领域。开发具有用户动态交互、富媒体应用的新一代 Web 网站需要 HTML、CSS、JavaScript、DOM、AJAX 等组合技术，其中 HTML、CSS、JavaScript 三大技术称为**“Web 标准三剑客”**。

## 1.3.1　HTML

HTML（Hypertext Markup Language）是超文本标记语言。它是一种标记语言，而不是编程语言。HTML 是 Web 页面的结构。HTML 使用标记来描述网页。网页的内容包括标题、副标题、段落、无序列表、定义列表、表格、表单等。

HTML 是 SGML（Standard Generalized Markup Language，标准通用标记语言）下的一个应用（也称为一个子集），也是一种标准规范，它通过标记符号来标记要显示的网页中的各个部分。而 SGML 是一种定义电子文档结构和描述其内容的国际标准语言，是所有电子文档标记语言的起源。

HTML 文档是用来描述网页，由 HTML 标记和纯文本构成的文本文件。Web 浏览器可以读取 HTML 文档，并以网页的形式显示出它们。例如在 Chrome 浏览器的 URL 中输入网址 http://www.edu.cn，所看到的网页就是浏览器对 HTML 文件进行解释的结果，如图 1-5 所示。




图 1-5　中国教育和科研计算机网首页

右击网页的任何位置，从弹出菜单中选择“查看网页源代码”，如图 1-6 所示。其中<head>、<meta>、<title>、<link>等都是 HTML 的标记，浏览器能够正确地理解这些标记，并呈现给用户。




图 1-6　中国教育和科研计算机网首页源代码

下面简单介绍 HTML 超文本标记语言的发展历史。

- HTML1.0：1993 年 6 月，互联网工程工作小组（IETF）发布工作草案；
- HTML2.0：1995 年 11 月，发布 RFC1866，在 RFC2854 于 2000 年 6 月发布之后被宣布已经过时；
- HTML3.2：1996 年 1 月 14 日发布，W3C 推荐标准；
- HTML4.0：1997 年 12 月 18 日发布，W3C 推荐标准；
- HTML4.01：1999 年 12 月 24 日发布，W3C 推荐标准；
- HTML5：2014 年 10 月 28 日发布，W3C 推荐标准。

## 1.3.2　CSS

由于 Netscape 和 Microsoft 两家公司在自己的浏览器软件中不断地将新的 HTML 标记和属性（例如字体标记和颜色属性）添加到 HTML 规范中，导致创建具有清晰的文档内容并独立于文档表现层的站点变得越来越困难。为了解决这个问题，Hakon Wium Lie（哈肯·维姆·莱，挪威）和 Bert Bos（伯特·波斯，荷兰）于 1994 年共同发明了级联样式表。

**1．CSS 的作用**

级联样式表（Cascading Style Sheet，CSS），也称为层叠样式表。在设计 Web 网页时采用 CSS 技术，可以有效地对页面的布局、字体、颜色、背景和其他效果实现更加精确的控制。只要对相应的代码做一些简单的修改，就可以改变同一页面的不同部分，或者同一个网站的不同页面的外观和格式。采用 CSS 技术是为了解决网页内容与表现分离的问题。

CSS 语言是一种标记语言，不需要编译，属于浏览器解释型语言，可以直接由浏览器解释执行。CSS 标准由 W3C 的 CSS 工作组制定和维护。

**2．CSS 的发展历史**

- CSS1：1996 年 12 月 17 日发布，W3C 推荐标准。
- CSS2：1999 年 1 月 11 日发布，W3C 推荐标准，CSS2 添加了对媒介（打印机和听觉设备）、可下载字体的支持。
- CSS3：计划将 CSS 划分为更小的模块，这些模块包括盒子模型、列表模块、超链

接方式、语言模块、背景和边框、文字特效、多栏布局等。

### 1.3.3　JavaScript

在 HTML 基础上，使用 JavaScript 可以开发交互式 Web 页面。JavaScript 的出现使得网页和用户之间实现了一种实时性的、动态的、交互性的关系，使网页包含更多活跃元素和更加精彩的内容。这也是 JavaScript 与 HTML DOM 共同构成 Web 网页的行为。

**1．JavaScript 由来**

JavaScript 是一种基于对象和事件驱动并具有相对安全性的客户端脚本语言。同时也是一种广泛用于客户端 Web 开发的脚本语言，常用来给 HTML 网页添加动态的功能，例如响应用户的各种操作。JavaScript 最初由 Netscape 的 Brendan Eich（布兰登 • 艾奇）设计，是一种由 Netscape 的 LiveScript 发展而来、原型化继承面向对象动态类型的客户端脚本语言，主要目的是为服务器端脚本语言提供数据验证的基本功能。在 Netscape 与 Sun 合作之后，LiveScript 更名为 JavaScript，同时 JavaScript 也成为原 Sun 公司的注册商标。**欧洲计算机制造商协会**（European Computer Manufacturers Association，ECMA）以 JavaScript 为基础制定了 ECMAScript 标准。

**2．JavaScript 的组成**

一个完整的 JavaScript 实现是由以下三个不同部分组成的：

- 核心（ECMAScript）。
- 文档对象模型（Document Object Model，DOM）。
- 浏览器对象模型（Browser Object Model，BOM）。

### 1.3.4　HTML DOM

HTML DOM（Document Object Model）是 HTML 文档对象模型的缩写。根据 W3C DOM 规范，DOM 是一种与浏览器、平台语言无关的接口，使得用户可以访问页面上其他的标准组件。DOM 与 JavaScript 结合起来实现了 Web 网页的行为与结构的分离。

**1．DOM 由来**

简单理解，DOM 解决了 Netscape 的 JavaScript 和 Microsoft 的 JScript 之间的冲突，为 Web 设计师和开发者提供了一个处理 HTML 或 XML 文档标准的方法，方便访问站点中的数据、脚本和表现层对象。

借助于 JavaScript 可以重构整个 HTML 文档，可以添加、移除、改变或重排页面上的元素。JavaScript 需要获得对 HTML 文档中所有元素进行访问的入口，这个入口连同对 HTML 元素进行添加、移动、改变或移除的方法和属性，都是通过文档对象模型 DOM 来获得的，HTML DOM 定义了访问和操作 HTML 文档的标准方法。

**2．HTML DOM Level**

- DOM Level 1：1998 年 10 月发布，W3C 推荐规范，含有 DOM Core 和 DOM HTML 两个模块。
- DOM Level 2：引入 DOM 视图、DOM 事件、DOM 样式、DOM 遍历和范围；用于处理新的接口类型。
- DOM Level 3：引入以统一的方式载入和保持文档的方法，包含在新模块 DOM Load

and Save 和 DOM Validation 方法，进一步扩展了 DOM。

### 1.3.5 BOM

BOM（Browser Object Model）也称浏览器对象模型。浏览器对象模型定义了 JavaScript 可以进行操作的浏览器的各个功能部件的接口，提供访问文档各个功能部件（如窗口本身、屏幕功能部件、浏览历史记录等）的途径以及操作方法。

IE 3.0 和 Netscape Navigator 3.0 浏览器提供了一个浏览器对象模型特性，可以对浏览器窗口进行访问和操作。使用 BOM，开发者可以移动窗口、改变状态栏中的文本以及执行其他与页面内容不直接相关的动作。由于没有相关的 BOM 标准，每种浏览器都有自己的 BOM 实现的方法。有一些事实上的标准，如具有一个窗口对象和一个导航对象，不过每种浏览器都可以为这些对象或其他对象定义自己的属性和方法。

BOM 主要处理浏览器窗口和框架，不过通常浏览器特定的 JavaScript 扩展都被看作 BOM 的一部分。这些扩展包括：

- 弹出新的浏览器窗口。
- 移动、关闭浏览器窗口以及调整窗口大小。
- 提供 Web 浏览器详细信息的定位对象。
- 提供用户屏幕分辨率详细信息的屏幕对象。
- 对 cookie 的支持。
- Internet Explorer 对 BOM 进行扩展以包括 ActiveX 对象类，可以通过 JavaScript 来实现 ActiveX 对象。

常见 BOM 对象有 Window 对象、Navigator 对象、Screen 对象、History 对象、Location 对象等。

### 1.3.6 AJAX

AJAX（Asynchronous JavaScript and XML）也称**异步 JavaScript 和 XML**，在 Web 2.0 的热潮中，已成为人们谈论最多的技术术语。AJAX 是多种技术的综合，它使用 XHTML 和 CSS 标准化呈现，使用 DOM 实现动态显示和交互，使用 XML 和 XSTL 进行数据交换与处理，使用 XMLHttpRequest 对象进行异步数据读取，使用 JavaScript 绑定和处理所有数据。更重要的是它打破了使用页面重载的惯例技术组合，可以说 AJAX 已成为 Web 开发的重要武器。

传统的网页（不使用 AJAX）如果需要更新内容，必须重载整个网页页面，而使用 AJAX 则可以部分更新网页内容。有很多使用 AJAX 的应用程序案例，如新浪微博、Google 地图、开心网等。

通过 AJAX，可以使用 JavaScript 的 XMLHttpRequest 对象来直接与服务器进行通信，不再需要重载页面与 Web 服务器交换数据。

AJAX 在浏览器与 Web 服务器之间使用异步数据传输（HTTP 请求），这样就可使网页从服务器请求少量的信息，而不是整个页面。

### 1.3.7 jQuery

jQuery 是一套跨浏览器的 JavaScript 库，简化 HTML 与 JavaScript 之间的操作。由 John Resig 在 2006 年 1 月的 BarCamp NYC 上发布第一个版本。目前是由 Dave Methvin 领导的开发团队进行开发。全球前 10000 个访问量最高的网站中，有 59%使用了 jQuery，它是目前最受欢迎的 JavaScript 库。

jQuery 由美国人 John Resig 创建，至今已吸引了来自世界各地的众多 JavaScript 高手加入其开发团队，包括来自德国的 Jorn Zaefferer、罗马尼亚的 Stefan Petre 等。jQuery 是继 prototype JS 框架之后又一个优秀的 JavaScript 框架。其宗旨是“Write Less,Do More”，即“写更少的代码，做更多的事情”。

## 1.4 Web 前端开发工具

在 HTML 基础上，使用 JavaScript 可以开发交互式 Web 页面。JavaScript 的出现使得网页和用户之间实现了一种实时性的、动态的、交互性的关系，使网页包含更多活跃元素和更加精彩的内容。而用于开发 Web 前端应用工具有很多，可以根据使用习惯进行选择。

### 1.4.1 EditPlus

EditPlus 是 Windows 下的一个文本、HTML、PHP 以及 Java 编辑器。它不但是记事本的一个很好的代替工具，同时它也为网页制作者和程序设计员提供了许多强大的功能。对 HTML、PHP、Java、C/C++、CSS、ASP、Perl、JavaScript 和 VBScript 的语法有突出显示。同时，根据自定义语法文件它能够扩展支持其他程序语言。无缝网络浏览器预览 HTML 页面，以及 FTP 命令上载本地文件到 FTP 服务器。EditPlus V4.0 软件程序界面如图 1-7 所示，其中 HTML 默认加载模板 tamplate.html 编码类型已经由 ANSI 改为 UTF-8。

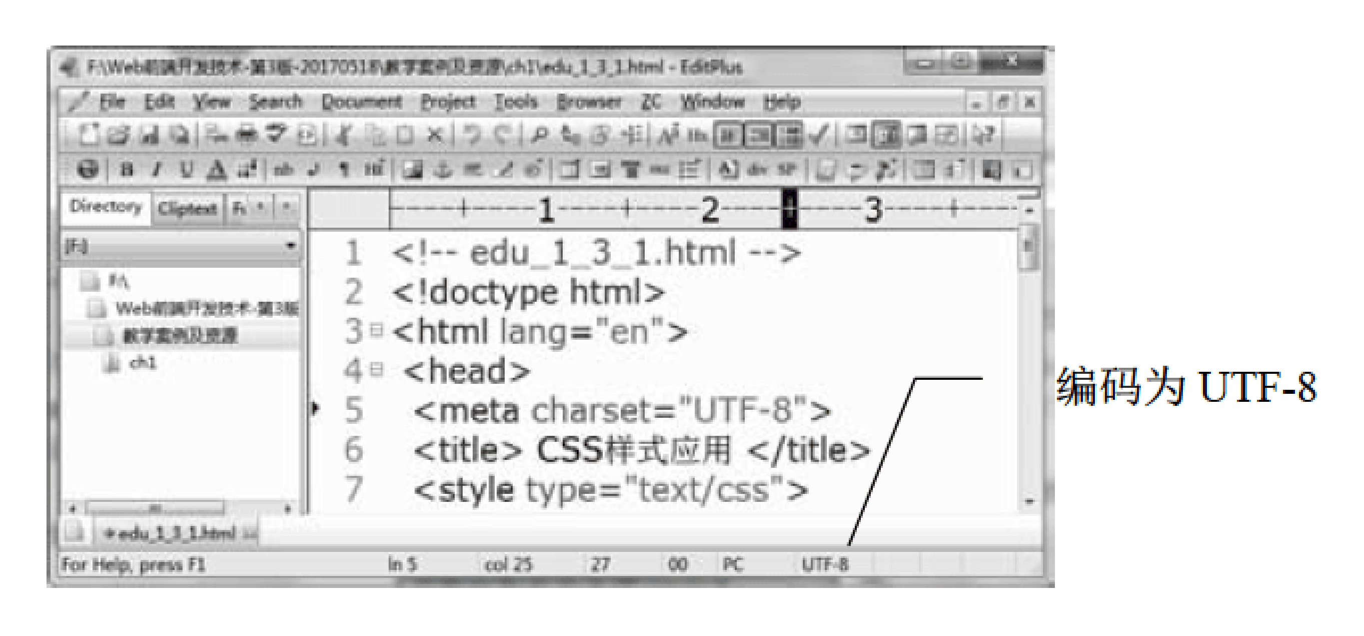



图 1-7　EditPlus 程序界面

### 1.4.2 Adobe Dreamweaver

Adobe Dreamweaver（前称 Macromedia Dreamweaver）是 Adobe 公司的著名网站开发工具，是集网页制作和管理网站于一身的所见即所得的网页编辑器，利用它可以轻而易举地制作出跨越平台限制和浏览器限制的充满动感的网页。程序界面如图 1-8 所示。

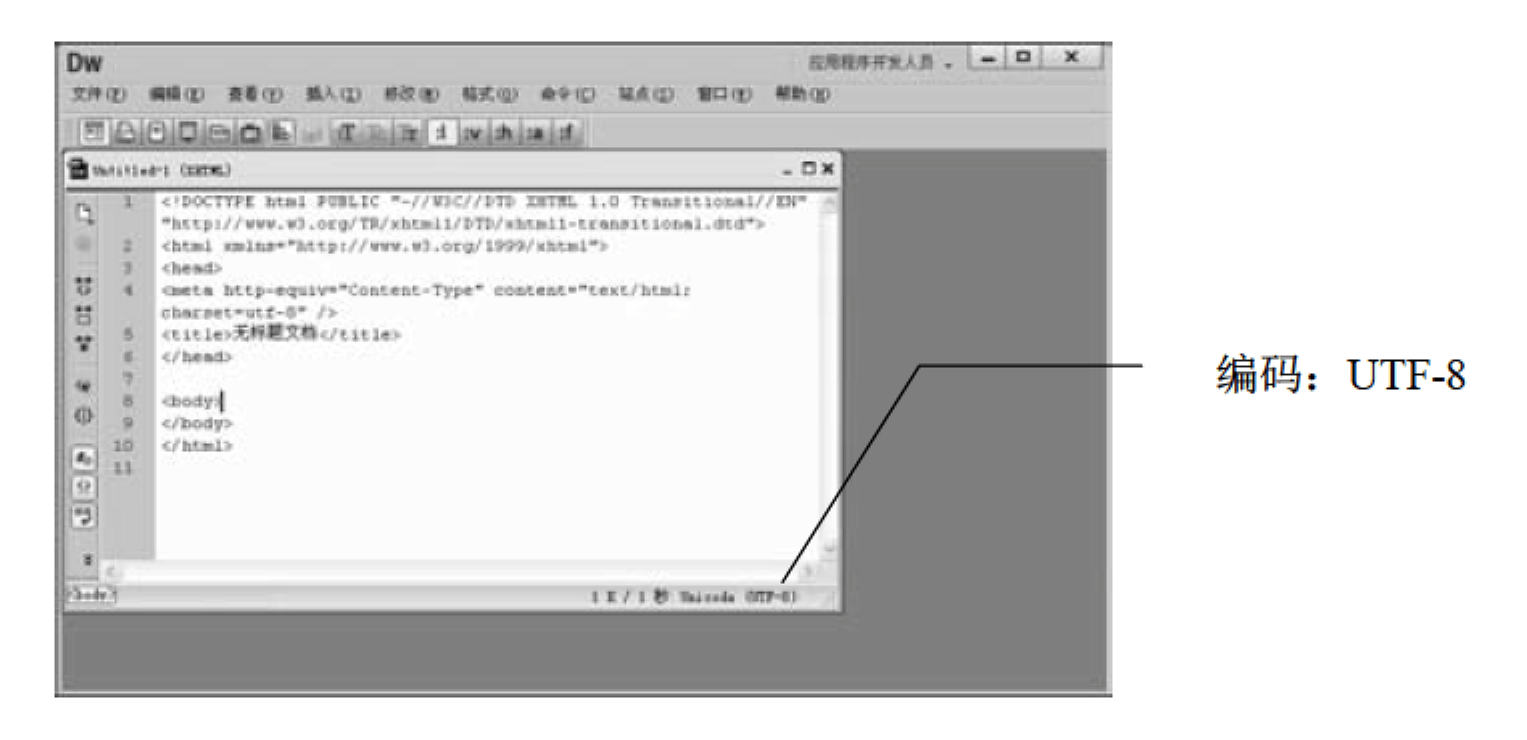



图 1-8　Dreamweaver 程序界面

目前有 Mac 和 Windows 系统的版本。Adobe Dreamweaver CS6 提供了一套直观的可视界面，可以创建和编辑 HTML 网站和移动应用程序。

## 1.4.3　Sublime Text

Sublime Text 支持多种编程语言的语法高亮、拥有优秀的代码自动完成功能，还拥有代码片段（Snippet）的功能，可以将常用的代码片段保存起来，在需要时随时调用；支持 VIM 模式、支持宏。使用此软件写代码，会提高编码的速度与效率。程序界面如图 1-9 所示。

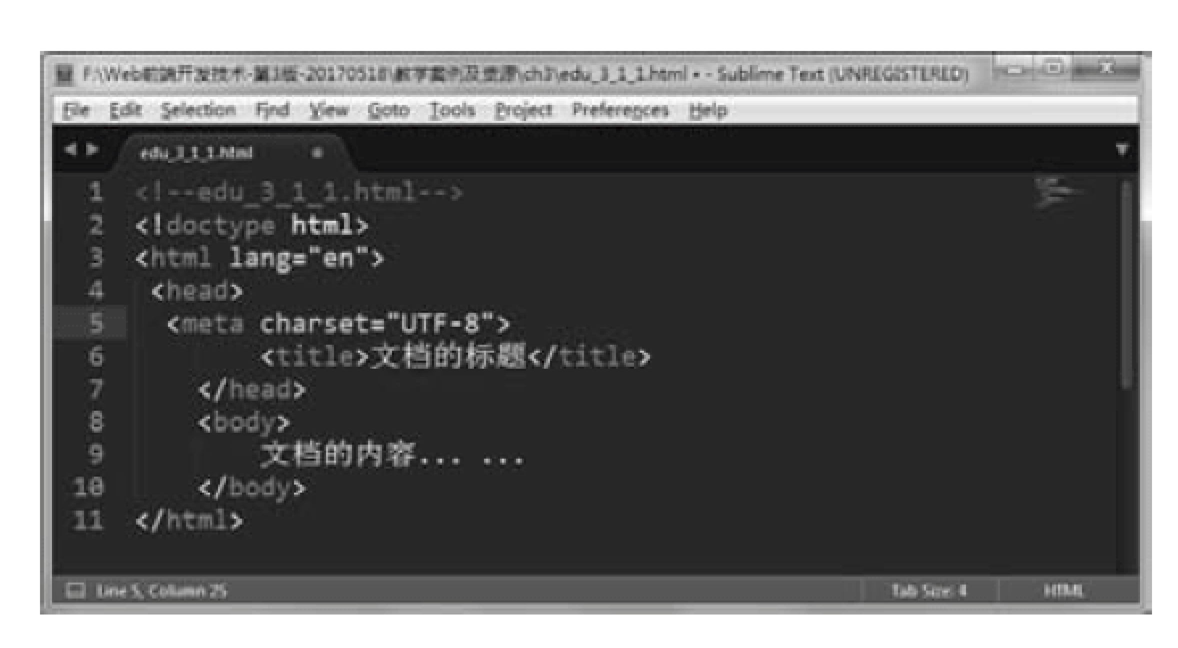


图 1-9　Sublime Text 程序界面

## 1.4.4　WebStorm

WebStorm 是 JetBrains 公司旗下的一款 JavaScript 开发工具。目前已经被广大中国 JS 开发者誉为 Web 前端开发神器、最强大的 HTML5 编辑器、最智能的 JavaScript IDE 等。与 IntelliJ IDEA 同源，继承了 IntelliJ IDEA 强大的 JS 部分的功能。程序界面如图 1-10 所示。



图 1-10　WebStorm 程序界面

### 1.4.5　HBuilder

HBuilder 是 DCloud（数字天堂）推出的一款支持 HTML5 的 Web 开发 IDE。HBuilder 的编写用到了 Java、C、Web 和 Ruby。HBuilder 本身主体是由 Java 编写，它基于 Eclipse，所以顺其自然地兼容了 Eclipse 的插件。快，是 HBuilder 的最大优势，通过完整的语法提示和代码输入法、代码块等，大幅提升 HTML、JS、CSS 的开发效率。程序界面如图 1-11 所示。

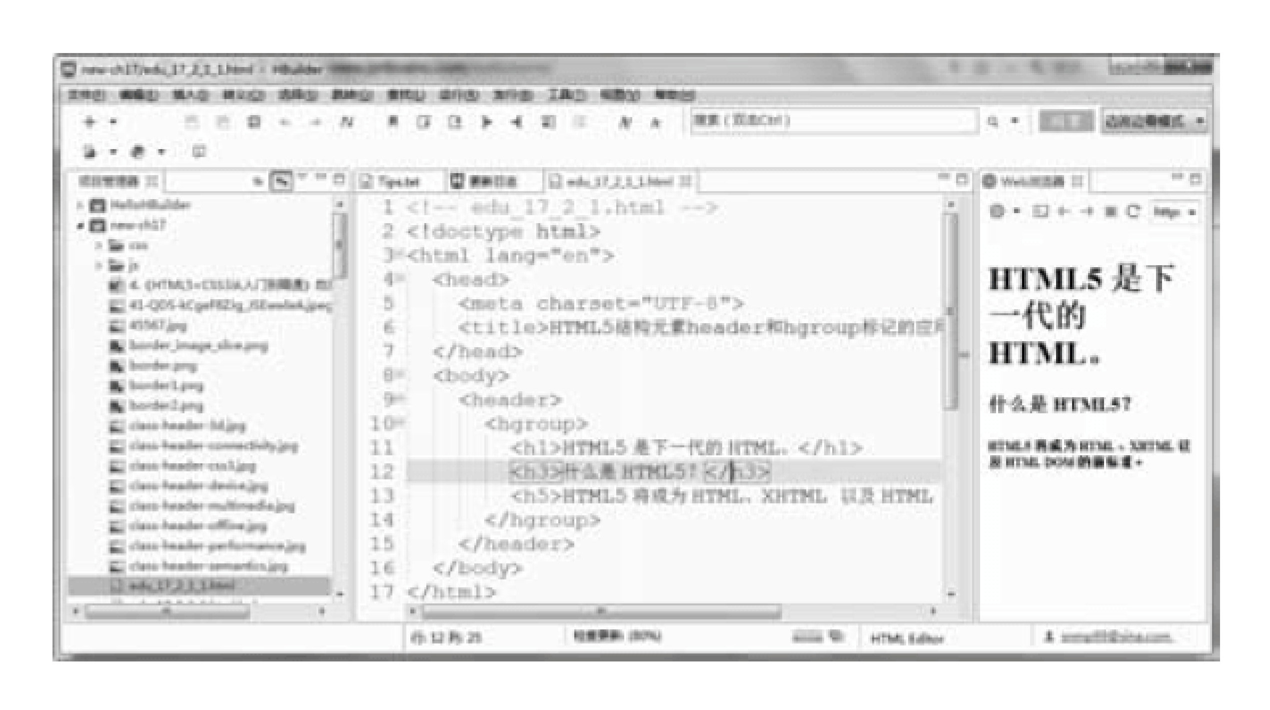



图 1-11　HBuilder 程序界面

## 1.5　浏览器工具

使用 HTML、CSS、JavaScript 组合技术设计的 Web 网站，需要经过发布，才能通过浏览器来观看其设计效果。基于 Internet 的各类网页浏览器有很多，据 StatCounter 和 Net Applications 两大市场研究公司统计分析，2017 年 10 月和 6 月全球浏览器市场份额统计结果表现：排名全球前五名的浏览器分别是 Google Chrome、Microsoft IE、Mozilla Firefox、Safari、Opera。由于两家公司计算方法不同，造成浏览器排名顺序不同。其市场占有份额分别如图 1-12 和图 1-13 所示。

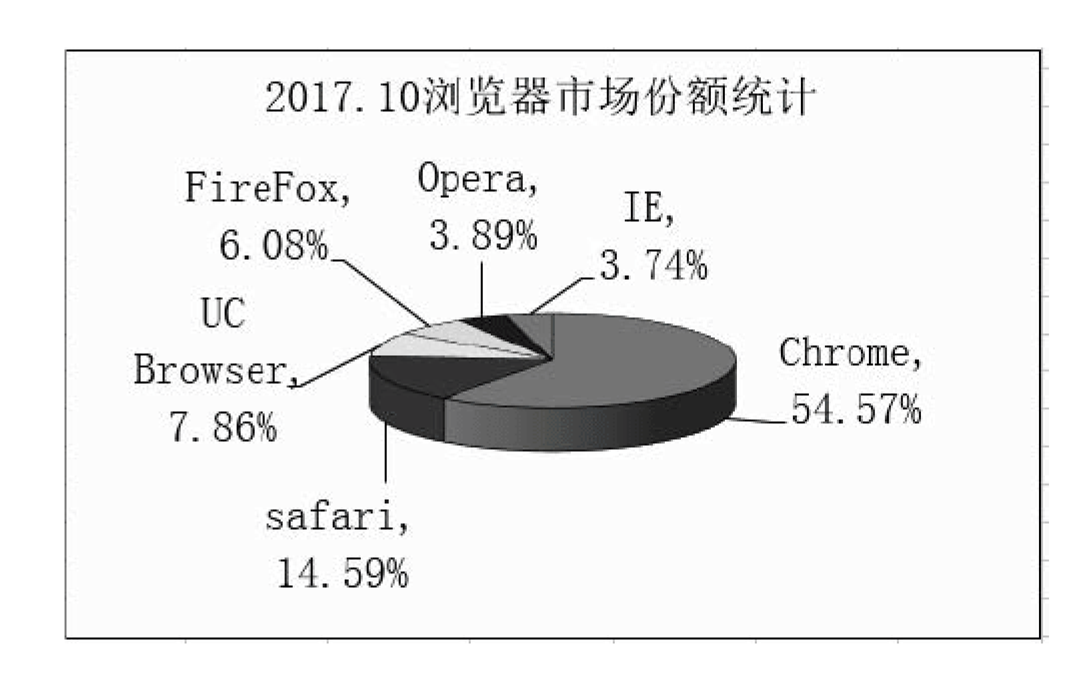



图 1-12　StatCounter 统计数据

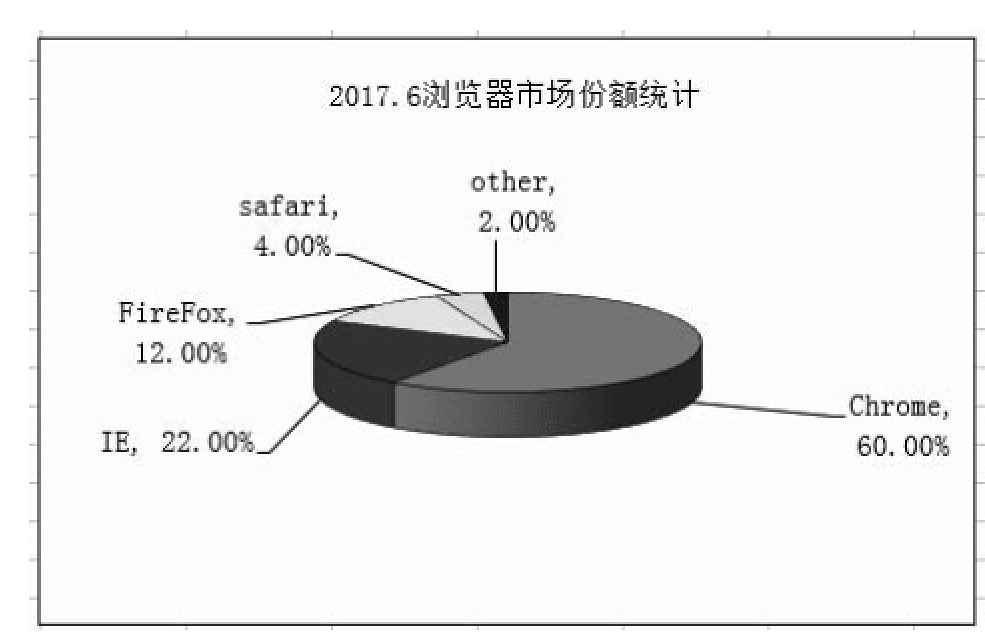



图 1-13　Net Applications 统计数据

各类浏览器对应的标识如图 1-14 所示。作为 Web 前端开发工程师一定要了解不同浏览器的使用性能和特点，了解它们的差异性，在编写 Web 网页代码时才能充分考虑到浏览器的兼容性，让网站在不同浏览器中显示效果与风格相同。



图 1-14　主流浏览器对应的标识

## 1.5.1　Internet Explorer

Internet Explorer 是微软公司推出的一款网页浏览器。虽然自 2004 年以来 Internet Explorer 丢失了一部分市场占有率，但依然是使用最广泛的网页浏览器。竞争对手主要有 Chrome、Firefox、Safari、Opera 等。目前最新版本是 IE 11.0，用户可根据自己的计算机配置选择安装相关版本的浏览器。

## 1.5.2　Google Chrome

Google Chrome，又称 Google 浏览器，是一个由 Google 公司开发的开源网页浏览器。该浏览器基于其他开源代码软件编写，包括 WebKit 和 Mozilla，目标是提升稳定性、速度和安全性，并创造出简单且高效的使用者界面。软件的名称来自于称作 Chrome 的网络浏览器图形用户界面（Graphic User Interface,GUI）。软件的 beta（测试）版本在 2008 年 9 月 2 日发布，提供 43 种语言版本，有支持 Windows、Mac OS X 和 Linux 版本并提供下载。Chrome 也成为使用最广泛的浏览器。

## 1.5.3　Mozilla Firefox

Mozilla Firefox 中文名通常称为“火狐”，是一个开源网页浏览器，使用 Gecko 引擎（即非 IE 内核），可以在多种操作系统如 Windows、Mac 和 Linux 上运行。Firefox 由 Mozilla 基金会与数百个志愿者所开发，原名“Phoenix”（凤凰），之后改名“Mozilla Firebird”（火鸟），再改为现在的名字，Firefox 的市场份额在全球荣居第三位。

## 1.5.4　Safari

Safari 是苹果计算机的最新操作系统 Mac OS X 中新的浏览器，用来取代之前的 Internet Explorer for Mac。Safari 使用了 KDE 的 KHTML 作为浏览器的计算核心。目前该浏览器已支持 Windows 平台，但是与运行在 Mac OS X 上的 Safari 相比，有些功能出现丢失。Safari 也是 iPhone 手机、iPod Touch、iPad 平板电脑 iOS 的指定默认浏览器。

## 1.5.5　Opera

Opera 浏览器是一款由挪威 Opera Software ASA 公司制作的支持多页面标签式浏览的网络浏览器，是跨平台的浏览器，可以在 Windows、Mac、FreeBSD、Solaris、BeOS、OS/2、QNX、Linux 等多种操作系统上运行。Opera 浏览器创始于 1995 年 4 月，到 2017 年 10 月，官方发布的个人计算机用的最新版本为 Opera48。此外，Opera 还有手机用的版本，如在 Windows Mobile 和 Android 手机上安装的 Opera Mobile 和 Opera Mini，也支持多语言，包括简体中文和繁体中文。

# 1.6 综合实例

以 Web 前端开发技术综合运用为例，介绍运用 HTML、CSS、JavaScript 三大技术实现 Web 网页设计。代码如下所示，其页面效果如图 1-15 所示。

视频讲解

```
1 <!-- edu_1_6_1.html -->
2 <!doctype html>
3 <html lang="en">
4    <head>
5        <meta charset="UTF-8">
6        <title>Web前端开发技术初步应用</title>
7        <style type="text/css">
8            p{font-size:20px;color:red;text-indent:2em;}
9            h3{font-size:24px;font-weight:bolder;color:#000099;}
10         </style>
11     </head>
12     <body>
13         <h3>Web前端开发技术</h3>
14         <p>HTML</p>
15         <p>CSS</p>
16         <p>JavaScript</p>
17         <h3>网络学习资源</h3>
18         <a href="http://www.w3school.com.cn/html/">HTML教程</a>
19         <script type="text/javascript">
20             alert ("Web前端开发工程师就业前景好、待遇高！");
21         </script>
22     </body>
23 </html>
```




图 1-15　Web 前端开发技术综合实例页面

上述代码中第 8 行定义段落 P 标记样式为字大小为 20px、颜色为红色、段落缩进 2 个字符；第 9 行定义 3 号标题字 h3 标记样式为字大小为 24px、字体粗细为特粗、颜色为 #000099；第 12～22 行是 HTML 的主体，包含标题字、段落、超链接、脚本标记的定义，其中第 13 行、第 17 行定义 h3 标题字，第 14～16 行定义 3 个段落 P 标记，第 18 行定义超链接 a 标志，第 19～21 行定义脚本 script 标记，在其中插入告警消息框 alert()输出信息

“Web 前端开发工程师就业前景好、待遇高！”。

## 本 章 小 结

本章从 Web 概述、Web 前端开发工程师职业要求、Web 前端开发技术、Web 前端开发工具、Web 浏览器五大方面对 Web 前端开发技术进行综述。

重点阐述了 Web 概述、Web 起源、Web 特点、Web 工作原理。为适应互联网行业迅速发展对 IT 开发人才的需要，介绍了 Web 前端开发工程师这一紧缺岗位的职业需求。

Web 前端开发技术重点介绍了 Web 网页设计的“三剑客”，分别是 HTML、CSS、JavaScript，三者在网页设计中作用各不相同。其中 HTML 是 Web 网页的内容；CSS 是 Web 网页的表现；JavaScript 和 HTML DOM 是网页的行为，实现网页的动态、交互的功能。AJAX 在浏览器与 Web 服务器之间使用异步数据传输，这样就可使网页从服务器请求少量的信息，而不是整个页面。jQuery 是一套跨浏览器的 JavaScript 库，简化 HTML 与 JavaScript 之间的操作。

Web 前端开发工具重点介绍了目前 Web 前端开发常用的工具。

Web 浏览器重点介绍各大主流网络浏览器，通过使用了解浏览器之间的差异。

## 练习与实验

### 练习 1

**1. 选择题**

（1）HTML 是一种（　　）语言。

A．编译型　　B．超文本标记　　C．高级程序设计　　D．面向对象的编程

（2）世界上第一个网页是（　　）。

A．http://www.w3c.org　　B．http://info.cern.ch

C．http://www.microsoft.com　　D．http://www.baidu.com

（3）访问 FTP 站点使用的协议类型是（　　）。

A．http　　B．ftp　　C．https　　D．mailto

（4）下列不是开发 HTML 网页的软件是（　　）。

A．EditPlus　　B．NotePad　　C．TextPad　　D．Visual BASIC

（5）设计 JavaScript 语言的公司是（　　）。

A．Netscape　　B．Microsoft　　C．Sun　　D．Google

**2. 填空题**

（1）HTML 文档是由______构成的______文件。

（2）世界上第一个网站的发明人是____________。

（3）从 IE 浏览器菜单中选择______命令，可以在打开的记事本中查看网页的源代码。

（4）列出常用的 Web 前端开发工具（三个以上）____________、____________、____________。

（5）HTML 的全称是__________________。URL 的全称是__________________。CSS 的全称是__________________。AJAX 的全称是__________________。

（6）列出常用的主流网络浏览器（三个以上）__________、__________、__________。

**3. 简答题**

（1）简述 Web 的工作原理。

（2）Web 具有哪些特点？

（3）写出 URL 的格式，并说明它的组成及作用。

（4）分别说明 HTML、CSS、JavaScript 在 Web 网页设计中的作用。

## 实验 1

1．学会使用 NotePad 和 EditPlus 等编辑软件将综合实例的代码输入到编辑器中，并进行调试，通过浏览器查看网页效果并与图 1-15 进行比较。

2．下载 HBuilder、Sublime Text、WebStorm 等软件，练习使用各种编辑器软件，试比较各自的优缺点。

# 第 2 章 HTML 基础

本章学习目标

**通过本章的学习，能够掌握 HTML 基本组成结构，理解 HTML 头部 head 和主体 body 两大部分在网页设计中的作用。理解 head、body 标记中可以包含哪些标记；理解 HTML 标记的作用及标记语法，理解标记的类型，学会编写简易的 Web 网页代码。**

**Web 前端开发工程师应掌握以下内容：**

**掌握 HTML 文档的基本结构。**

**理解标记类型、标记语法。**

**学会 body 标记属性的设置方法。**

**学会给网页添加注释。**

**理解元信息 meta 标记的作用。**

## 2.1 HTML 文档结构

视频讲解

HTML 文档由头部 head 和主体 body 两个部分组成。头部 head 标记中可以定义标题、样式等，头部信息不显示在网页上；主体 body 标记中可以定义段落、标题字、超链接、脚本、表格、表单等元素，主体内容是网页要显示的信息。

HTML 文档基本结构

```
<html lang="en">
 <head>
     <meta charset="UTF-8">
     <meta name="Keywords" content="">           这是头部
     <meta name="Description" content="">
     <title>Web网页标题</title>
 </head>
 <body>
    …                                  这是主体
 </body>
</html>
```

【例 2-1-1】HTML 文档的基本结构展示。页面效果如图 2-1 所示。

```
<!--
    程序名称：edu_2_1_1.html
    程序功能：HTML文档结构
    设计人员：Web前端开发工程师
    设计时间：2017/5/31
```

```
 6 -->
 7 <!doctype html>
 8 <html lang="en">
 9     <head>
10         <meta charset="UTF-8">
11         <title> HTML文档结构 </title>
12         <style type="text/css">
13             p{font-size:24px; /* 定义字体大小 */ }
14         </style>
15     </head>
16     <body>
17         <p>HTML文档结构由head、body标记组成</p>
18         <h3>标题字h3</h3>
19         <hr size=3 color="red">
20         <a href="http://www.baidu.com">百度</a>
21         <script type="text/javascript">
22             document.write("这是简单的网页！"); //向页面输出信息
23         </script>
24     </body>
25 </html>
```

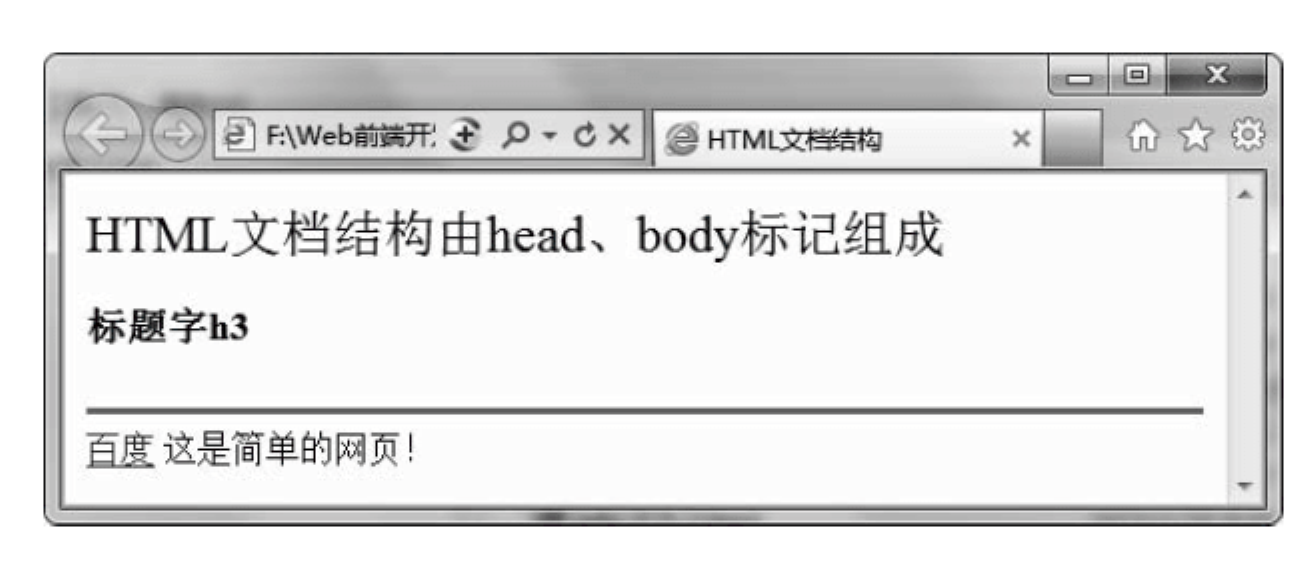



图 2-1　HTML 文档结构展示

HTML 文档以<html>标记开始，以</html>标记结束。所有的 HTML 代码都位于这两个标记之间。浏览器根据 HTML 文档类型和内容来解释整个网页，然后呈现给用户。一般情况下，每个 HTML 文档都应该有且只有一个 html、head 和 body 元素。

**代码解释**

代码中第 9～15 行是头部标记所包含的代码，头部标记所包含的内容不会在网页上显示；第 16～24 行是主体标记所包含的代码，也是网页要显示的主要信息。

# 2.2　头部 head

HTML 文档的头部 head 标记主要包含页面标题标记、元信息标记、样式标记、脚本标记、链接标记等。头部 head 标记所包含的信息一般不会显示在网页上。

## 2.2.1　标题 title 标记

基本语法

```
<title>标题信息显示在浏览器的标题栏上</title>
```

语法说明

title 标记是成对标记，<title>是开始标记，</title>是结束标记，两者之间的内容为显示在浏览器的标题栏上的信息。

【例 2-2-1】标题 title 标记应用。代码如下所示，页面效果如图 2-2 所示。

视频讲解

```
<!-- edu_2_2_1.html  -->
<!doctype html>
<html lang="en">
    <head>
        <meta charset="UTF-8">
        <title> 页面标题 </title>
    </head>
    <body>
        页面标题显示在浏览器的标题栏上
    </body>
</html>
```




标题信息显示在浏览器标题栏

图 2-2　标题 title 标记应用

## 2.2.2　元信息 meta 标记

meta 标记用来描述一个 HTML 网页文档的属性，也称为元信息（meta-information），这些信息并不会显示在浏览器的页面中，例如作者、日期和时间、网页描述、关键词、页面刷新等。meta 标记是单个标记，位于文档的头部，其属性定义了与文档相关联的“名称/值”对。

### 1．meta 标记

基本语法

```
<meta name="" content="">
<meta http-equiv="" content="">
```

属性说明

meta 属性主要分为两组：

name 属性与 content 属性。

name 属性用于描述网页，它是“名称/值”形式中的名称，name 属性的值所描述的内容通过 content 属性表示，便于搜索引擎机器人查找、分类。其中最重要的是 description、keywords 和 robots。

http-equiv 属性与 content 属性。

http-equiv 属性用于提供 HTTP 协议的响应头报文，它回应给浏览器一些有用的信息，以帮助正确和精确地显示网页内容。它是“名称/值”形式中的名称，http-equiv 属性的值所描述的内容通过 content 属性表示。meta 标记的属性、取值和说明如表 2-1 所示。

表 2-1　meta 标记属性、取值及说明表

| 属　　性 | 值 | 说　　明 |
|---|---|---|
| name | author | 定义网页作者 |
| | description | 定义网页简短描述 |
| | keywords | 定义网页关键词 |
| | generator | 定义编辑器 |
| http-equiv | content-type | 内容类型 |
| | expires | 网页缓存过期时间 |
| | refresh | 刷新与跳转（重定向）页面 |
| | set-cookie | 如果网页过期，那么存盘的 cookie 将被删除 |
| content | some_text | 定义与 http-equiv 或 name 属性相关的元信息 |

### 2．meta 标记的使用方法

1）name 属性设置

```
<meta name="keywords" content="信息参数" />
<meta name="description" content="信息参数" />
<meta name="author" content="信息参数"/>
<meta name="generator" content="信息参数" />
<meta name="copyright" content="信息参数">
<meta name="robots" content="信息参数">
```

robots 告诉搜索引擎机器人抓取哪些页面。其属性取值及说明如表 2-2 所示。

表 2-2　robots 属性值及说明表

| 值 | 说　　明 |
|---|---|
| all | 文件将被检索，且页面上的链接可以被查询 |
| none | 文件将不被检索，且页面上的链接不可以被查询 |
| index | 文件将被检索 |
| noindex | 文件将不被检索，但页面上的链接可以被查询 |
| follow | 页面上的链接可以被查询 |
| nofollow | 文件将被检索，但页面上的链接不可以被查询 |

2）http-equiv 属性设置

```
1 <meta http-equiv="cache-control " content="no-cache">;
2 <meta http-equiv="refresh" content="时间; url=网址参数">
3 <meta http-equiv="content-type" content="text/html; charset=信息参数" />
4 <meta http-equiv="expires" content="信息参数" />
```

第 1 行说明禁止浏览器从本地计算机的缓存中访问页面内容，同时访问者将无法脱机浏览。第 2 行说明多长时间网页自动刷新，加上 URL 中的网址参数就代表多长时间自动链接其他网址。第 3 行中的 content-type 代表的是 HTTP 协议的头部，它可以向浏览器传回一些有用的信息，以帮助正确和精确地显示网页内容，与之对应的属性值为 content，content 中的内容其实就是各个参数的变量值。第 4 行设置 meta 标记的 expires（期限），可以用于设定网页在缓存中的过期时间。一旦网页过期，必须到服务器上重新传输。网页到期时间设置格式如下所示：

```
<meta http-equiv="expires" content="Fri 12 Jan 2001 18:18:18 GMT">
```

**注意：必须使用 GMT 的时间格式，或直接设为 0（数字表示多少时间后过期）。**

在 HTML5 规范和新版本软件中，第 3 行 meta 标记已经改为下列简洁形式：

```
<meta charset="UTF-8">
```


视频讲解

**【例 2-2-2】**元信息 meta 标记的应用，代码如下所示。

```
1 <!-- edu_2_2_2.html -->
2 <!doctype html>
3 <html lang="en">
4    <head>
5        <title>中国教育和科研计算机网CERNET</title>
6        <meta charset="UTF-8">
7        <meta content="IE=EmulateIE7" http-equiv="X-UA-Compatible">
8        <meta name="keywords" content="中国教育网,中国教育,科研发展,教育信息
         化,CERNET,CERNET2,下一代互联网,人才,人才服务,教师招聘,教育资源,教育服
         务,教育博客,教育黄页,教育新闻,教育资讯" />
9        <meta name="description" content="中国教育网（中国教育和科研计算机网）
         是中国最权威的教育门户网站，是了解中国教育的对内、对外窗口。网站提供关于中国
         教育、科研发展、教育信息化、CERNET等新闻动态、最新政策，并提供教师招聘、高
         考信息、考研信息、教育资源、教育博客、教育黄页等全面多样的教育服务。" />
10       <meta name="copyright" content="www.edu.cn" />
11       <meta name="robots" content="all" />
12   </head>
13   <body>
14       <p>这是中国教育和科研计算机网的头部部分标记的应用</p>
15   </body>
16 </html>
```

**代码解释**

代码是参照“中国教育和科研计算机网”的首页部分代码改写而成的。通过此段代码的示范让读者能够掌握在页面中如何正确地使用 meta 标记。代码中第 2 行是定义 HTML 文档类型（HTML5）；第 8～11 行使用 meta 标记定义了属性 name 的值分别为 keywords、description、copyright、robots 及相应的 content 的属性值。

# 2.3　主体 body

主体 body 是一个 Web 页面的主要部分，其设置内容是读者实际看到的网页信息。所有 WWW 文档的主体部分都是由 body 标记定义的。在主体 body 标记中可以放置网页中所有的内容，如图片、图像、表格、文字、超链接等元素。

## 2.3.1　body 标记

### 1. 基本语法

```
<body>
    这是网页的内容...
</body>
```

**2. 语法说明**

<body>是开始标记，</body>是结束标记。两者之间所包括的内容为网页上显示的信息。

【**例 2-3-1**】在 body 标记中插入相关标记。代码如下所示，页面效果如图 2-3 所示。




图 2-3　主体 body 标记的应用

视频讲解

```
1  <!-- edu_2_3_1.html -->
2  <!doctype html>
3 <html lang="en">
4  <head>
5   <meta charset="UTF-8">
6      <title>简易网页设计</title>
7      <style type="text/css">
8          p{text-indent:2em;/* 首行缩进2个字符 */}
9      </style>
10     </head>
11     <body text="green">
12         <h3 align="center">Web前端开发技术课程简介</h3>
13         <hr color="red">
14         <p>《Web前端开发技术》课程是计算机科学与技术、信息管理与信息系统、软件工
           程等专业的一门基础课程，也是其他计算机相关专业的公共基础课程，通过对Web前
           端开发三大主流技术学习和研究，让学生理解和掌握HTML、JavaScript、CSS等
           相关知识，通过实验培养学生设计与开发Web站点的基本操作技能。</p>
15     </body>
16  </html>
```

**3. 代码解释**

代码中第 11～15 行是主体部分；其中第 12 行是插入 h3 标记修饰标题；第 13 行是插入水平分隔线标记并设置成红色；第 14 行是插入一个段落 p 标记介绍课程。

## 2.3.2　body 标记属性

设置 body 标记属性可以改变页面的显示效果。该标记主要属性有 topmargin、leftmargin、text、bgcolor、background、link、alink、vlink。HTML5 中可以使用 CSS 属性替代。

**1. 基本语法**

```
<body leftmargin="50px" topmargin="50px" text="#000000" bgcolor="#339999"
link="blue" alink="white" vlink="red" background="body_image.jpg">
```

**2. 属性说明**

body 标记属性、取值及说明如表 2-3 所示。

表 2-3　body 标记属性、取值及说明表

| 属　　性 | 值 | 说　　明 |
|---|---|---|
| text | rgb(r,g,b)<br>rgb(r%,g%,b%)<br>#rrggbb 或#rgb<br><br>colorname | rgb 函数（整数），r、g、b 取值范围为 0～255。<br>rgb 函数（百分比），r、g、b 的值范围为 0～100。<br>十六进制数据（6 位或 3 位），如#rrggbb 或#rgb，r、g、b 为十六进制数，取值范围：0～9、A～F。例如#3F0，可转换为#33FF00。<br>颜色的英文名称，如 red、green、blue 等 |
| bgcolor | 同上 | 规定文档的背景颜色。不赞成使用 |
| alink | 同上 | 规定文档中活动链接的颜色。不赞成使用 |
| link | 同上 | 规定文档中未访问链接的默认颜色。不赞成使用 |
| vlink | 同上 | 规定文档中已被访问链接的颜色。不赞成使用 |
| background | URL | 规定文档的背景图像。不赞成使用 |
| topmargin | pixel | 规定文档中上边距的大小 |
| leftmargin | pixel | 规定文档中左边距的大小 |

【例 2-3-2】主体 body 标记属性的应用。代码如下所示，其页面效果如图 2-4 所示。

```
1 <!-- edu_2_3_2.html -->
2 <!doctype html>
3 <html lang="en">
4    <head>
5        <meta charset="UTF-8">
6        <title> body属性应用 </title>
7        <meta name="Generator" content="EditPlus">
8        <meta name="Author" content="储久良">
9        <style type="text/css">
10           div{background:#99cccc;width:500px;height:150px;}
11       </style>
12   </head>
13   <body text="rgb(00,00,00)" bgcolor="#f0f0f0"  background="" link=
     "rgb(0%,100%,0%)" alink="white" vlink="red" topmargin="60px"
     leftmargin="60px" >
14       <div id="" class="">
15           <p>欢迎访问我们的站点，我们为您提供网站地图。</p>
16           网站导航：
17           <a href="http://www.baidu.com">百度</a>
18           <a href="http://www.163.com">网易</a>
19           <a href="http://www.sina.com.cn">新浪</a>
20           <a href="http://www.sohu.com.cn">搜狐</a>
21       </div>
22   </body>
23 </html>
```

视频讲解

### 3. 代码解释

代码中第 10 行定义 div 的背景、宽度和高度；第 13 行设置 body 属性，其中设置网页信息显示的颜色为黑色，背景色为#f0f0f0，链接的颜色为绿色，活动链接为白色，访问过链接为红色，网页中的文档左边距、上边距均为 60px；第 15 行插入 1 个段落；第 17～20 行插入 4 个超链接。




图 2-4　主体 body 标记属性的应用

# 2.4　HTML 基本语法

HTML 文档结构主要由若干标记构成，随着页面复杂程度的不同，所使用的标记数量和标记属性设置也不相同。掌握 HTML 标记语法和标记属性语法是设计 Web 页面的基础。

## 2.4.1　标记类型

HTML 标记是由尖括号包围的关键词，用于说明指定内容的外貌和特征，也可称为标签（Tag），本书统一约定为标记。<html></html>、<head></head>、<body></body>、<br>、<hr>等都是标记。标记类型通常分为单（个）标记和双（成对）标记两种类型。

**1. 单（个）标记**

仅使用单个标记就能够表达特定的意思，称为单（个）标记。W3C 定义的新标准（XHTML1.0/HTML4.01）建议单个标记应以“/”结尾，即<标记名称/>。

1）基本语法

```
<标记名称>或<标记名称/>
```

2）语法说明

最常用的单个标记有<br>、<hr>、<link>。<br>、<br/>表示换行，<hr>、<hr/>表示水平分隔线，<link>表示链接标记。

**2. 双（成对）标记**

HTML 标记通常是**成对**出现的，比如<b>和</b>。标记对中的第一个标记是开始标记（也称为首标记），第二个标记是结束标记（也称为尾标记）。

1）基本语法

```
<标记名称>内容</标记名称>
```

2）语法说明

内容：是被成对标记说明特定外貌的部分。

例如，<html>与</html>之间的文本描述网页。<body>与</body>之间的文本是可见的页面内容。<strong>表示重要文本</strong>标记让浏览器将内容“表示重要文本”以标准粗体方式显示。

标记可以相互嵌套，但是不能交叉。尽管浏览器能够理解，但不是好的编程习惯。

例如，将 h3 标记与 i 标记交叉了，错误代码如下所示：

```
<h3><i>这是错误的交叉嵌套的代码</h3></i>        <!-- 交叉嵌套错误 -->
```

正确代码如下所示：

```
<h3><i>这是正确嵌套不交叉的代码</i></h3>
```

### 2.4.2 HTML 属性

HTML 使用标记来描述网页，浏览器根据标记解释标记所包含内容的效果。每一个标记均定义了一个默认的显示效果，这些默认效果是通过标记的附加信息（也称为属性 Attribute）来定义的。如果要修改某一个效果，那就需要修改该标记附加信息。

例如，段落 p 标记默认内容是居左对齐，如果需要将段落居中对齐显示，只需要设置对齐 align 属性。代码如下所示：

```
<p align="center">这个段落居中显示</p>
```

**1. 基本语法**

```
<标记名称 属性名1="属性值1" 属性名2="属性值2" … 属性名n="属性值n"></标记名称>
```

**2. 语法说明**

属性应在开始标记（首标记）内定义，且与首标记名称之间至少留有一个空格。例如，在上例的 p 标记中，align 为属性，center 为属性值，属性与属性值之间通过赋值号“=”连接，属性值可以直接书写，也可以使用双引号（“”）括起来。多个属性/值对之间至少留一个空格。

作为 Web 前端开发工程师应该养成一个良好的编写属性/值对的习惯，建议统一为属性值加上双引号，即：

```
属性名n="属性值n"
```

下列写法也是正确的：

```
<p align=center>这个段落居中显示</p>
```

**【例 2-4-1】**标记语法及属性语法的应用。代码如下所示，页面效果如图 2-5 所示。

视频讲解

```
<!-- edu_2_4_1.html -->
<!doctype html>
<html lang="en">
 <head>
  <meta charset="UTF-8">
  <title>标记语法及属性语法应用 </title>
  <style type="text/css">
      h2{text-align:center;background:#6699ff;padding:20px;}
      p{text-indent:2em;}
  </style>
 </head>
 <body background="" text="red">
```

```
13        <h2 align="center">新 年 寄 语</h2>
14        <hr size="2" color="#6600ff" width="100%"/>
15        <p align="left">轻轻送上我忠诚的祈求和祝愿，祈求分别的时光像流水瞬间逝去，
          祝愿再会时，紧握的手中溢满友情和青春的力量。 </p>
16        <p align="right">有一种跌倒叫站起，有一种失落叫收获，有一种失败叫成功——坚
          强些，朋友，明天将属于你！</p>
17    </body>
18 </html>
```

**3. 代码解释**

代码中第 8～9 行分别定义标题字 h2 样式（对齐、背景和填充等属性）、p 样式（首行缩进 2 个字符）；第 12～17 行是 HTML 的主体，包含标题字、水平分隔线、段落等标记定义，其中第 12 行定义主体 body 的所有文本信息的颜色属性，第 13 行定义标题字 h2 对齐 align 属性（居中对齐），第 14 行定义水平分隔线 hr 的粗细、颜色、宽度等属性，第 15 行定义段落 p 的对齐 align 属性（左对齐），第 16 行定义段落 p 的对齐 align 属性（右对齐）。

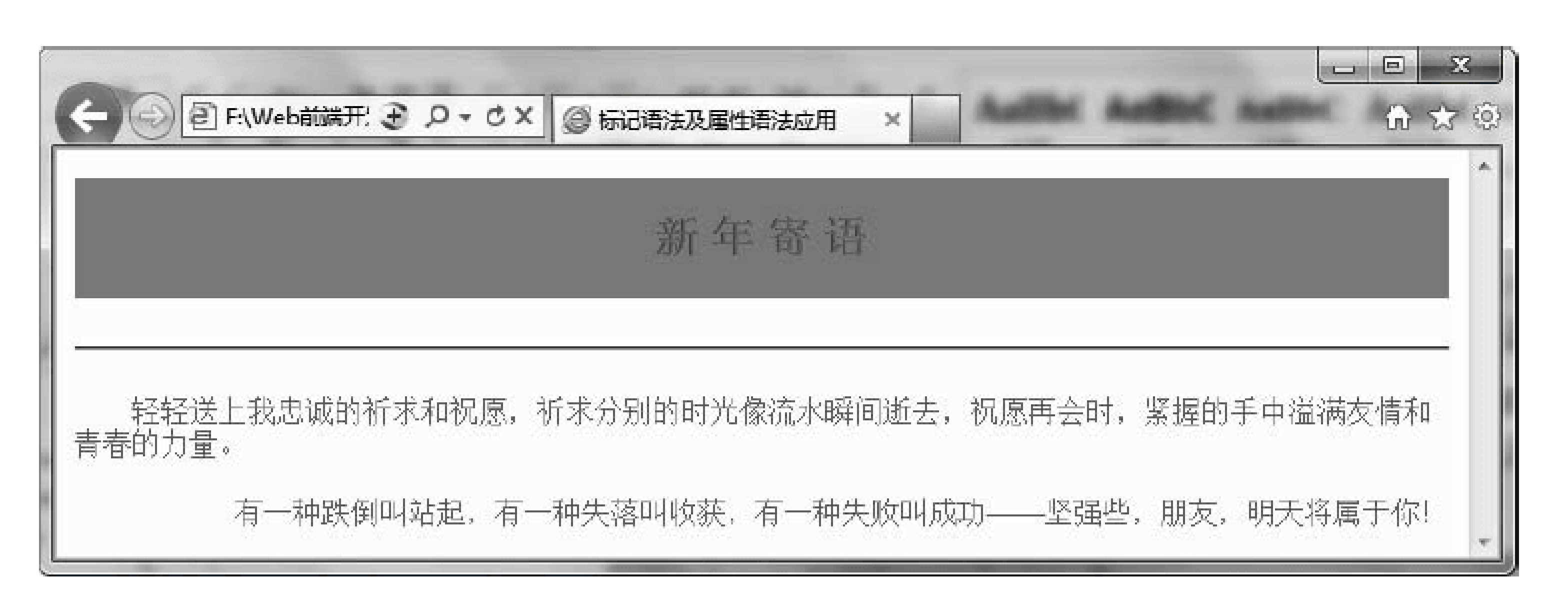



图 2-5　标记语法及属性语法应用

## 2.5　注　　释

为了提高代码的可读性、可维护性，作为 Web 前端开发工程师必须养成良好的编程习惯。通过注释标记给脚本代码或样式定义增加注释文本信息，可以给 Web 编程人员阅读和理解代码提供帮助，对后期软件维护和升级奠定基础。使用锯齿格式编写代码，即代码向右缩进 4 个字符，也可自定义缩进量。

在 HTML 代码中插入注释标记可以提高代码的可读性。浏览器不会解释注释标记，注释标记的内容也不会显示在页面上。

HTML 代码中添加注释的方法有两种：

<!-- 注释信息 -->。

<comment>注释信息</comment>。

**1.** <!-- 注释信息 -->

1）基本语法

```
<!-- 显示一个段落 -->
```

2）语法说明

以左尖括号和感叹号组合开始（<!--），以右尖括号（-->）结束。

**2. 注释 comment 标记**

1）基本语法

```
<comment>显示一个段落</comment>
```

2）语法说明

comment 标记是成对标记，以<comment>开始，以</comment>结束。标记包围的信息为注释内容。但这种方式很多浏览器（Chrome 等）会显示在页面上，所以不建议采用。

【例 2-5-1】给网页添加注释。代码如下所示，页面效果如图 2-6 所示。

视频讲解




图 2-6　添加注释应用

```
 1 <!-- edu_2_5_1.html -->
 2 <!doctype html>
 3 <html lang="en">
 4     <head>
 5         <meta charset="UTF-8">
 6         <title> 注释应用 </title>
 7     </head>
 8     <body>
 9         <comment>显示一个段落</comment>
10         <p>这是一个段落</p>
11         <script type="text/javascript">
12             document.write("HTML注释的应用");
13         </script>
14     </body>
15 </html>
```

**代码解释**

代码中第 1 行采用第 1 种注释方式；第 9 行采用第 2 种注释方式（IE 不会显示在页面上）；第 11～13 行是在 body 标记中插入脚本标记，第 12 行是向页面输出信息。

# 2.6　HTML 文档编写规范

HTML 文档是 Web 网页的重要文本文件，也是 Web 前端开发工程师设计网站的重要信息载体。文档编写质量直接影响网站呈现形式、访问速度、网络流量和用户体验，所以遵循 HTML 文档编写规范十分重要。

## 2.6.1　HTML 代码书写规范

HTML 语法是 Web 页面设计所应遵循的基本规范，养成按规范编写代码的习惯，能够大大减少设计页面中存在的缺陷。下面是 HTML 页面编码时需要注意的基本规范：

（1）HTML 标记是由尖括号包围的关键词。所有标记均以“<”开始、以“>”结束。结束的标记在开始标记名称前加上斜杠“/”。例如头部标记格式如下所示：

```
<head> … </head>
```

（2）根据标记类型，正确书写标记，单个标记最好在右尖括号前加 1 个斜杠“/”，如换行标记是单个标记<br/>，成对标记最好同时输入开始标记和结束标记，以免忘记。

（3）标记可以相互嵌套（也称为包含），但不能交叉。如：

```
<head><title> … </title> </head>    <!-- 这是正确的书写格式 -->
<head><style> …</head></style>    <!-- 这是错误的书写格式 -->
```

（4）在 HTML 代码书写时不区分大小写，如头部标记写成<HEAD>、<head>、<Head>、<HEAd>都可以，但建议在同一个 Web 开发项目保持一种风格，如统一小写标记名称。

（5）代码中包含任意多的回车符和空格在 HTML 页面显示时均不起作用。需要时可使用<br/>和 来实现换行和插入空格。为了代码清晰，建议不同的标记都单独占一行。

（6）给标记设置属性时，属性值建议用双引号标注起来。如段落内容居中格式如下所示：

```
<p align="center"> 这是段落信息居中显示 </p>
```

（7）书写开始与结束标记时，在左尖括号与标记名或与斜杠“/”之间不能留有多余空格，否则浏览器不能识别该标记，导致错误标记直接显示在页面上，影响页面美观效果。例如将【例 2-5-1】中第 9 行改成如下格式，错误的标记被显示在页面上，如图 2-7 所示。

```
< comment>显示一个段落< /comment>
```




图 2-7　添加注释应用

（8）编写 HTML 代码时，应该使用锯齿结构，即向右缩进 2～4 个字符，使代码结构清晰，提高代码的可读性，为后期阅读和维护提供帮助。

## 2.6.2　HTML 文档命名规则

HTML 文档是展示 Web 前端开发工程师成果的最好表示方式，为了便于文档规范化管理，在编写 HTML 文档时，必须遵循 HTML 文件命名规则。

HTML 文档命名规则如下：

（1）文档的扩展名为 htm 或者 html，建议统一用 html。

（2）文档名称只可由英文字母、数字或下画线组成，建议以字母或下画线开始。

（3）文档名称中不能包含特殊符号，如空格、$、&等。

（4）文档名称区分大小写，特别在 UNIX、Linux 系统中大小写表示的文件是不同的。

（5）Web 服务器主页一般命名为 index.html 或 default.html。

# 2.7　HTML 文档类型

Web 世界中存在许多不同的文档。只有了解文档的类型，浏览器才能正确地显示文档。HTML 也有多个不同的版本，只有完全明白页面中使用的确切 HTML 版本，浏览器才能完全正确地显示出 HTML 页面。

## 2.7.1　<!DOCTYPE>标记

DOCTYPE 是 Document Type 的英文缩写，<!DOCTYPE>标记不是 HTML 标记。此标记可告知浏览器文档使用哪种 HTML 或 XHTML 规范。<!DOCTYPE>声明位于文档中的最前面的位置，处于<html>标记之前。

基本语法

```
<!DOCTYPE element-name DTD-type DTD-name DTD-url>
```

语法说明

<!DOCTYPE>表示开始声明文档类型定义（Document Type Definition，DTD），其中 DOCTYPE 是关键字。

element-name 指定该 DTD 的根元素名称。

DTD-type 指定该 DTD 类型。设置为 PUBLIC，则表示该 DTD 是标准公用的，设置为 SYSTEM，则表示私人制定的。

DTD-name 指定该 DTD 的文件名称。

DTD-url 指定该 DTD 文件所在的 URL 地址。

>是指结束 DTD 的声明。

## 2.7.2　DTD 类型

**1．HTML4.01 的 DTD 定义（XHTML1.0 的 DTD 定义与此类似，省略）**

HTML4.01 中规定了三种 DTD 类型：严格型（Strict）、过渡型（Transitional）以及框架型（Frameset）。

1）HTML Strict DTD

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01//EN"
"http://www.w3.org/TR/html4/strict.dtd">
```

2）HTML Transitional DTD

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
"http://www.w3.org/TR/html4/loose.dtd">
```

3）HTML Frameset DTD

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Frameset//EN"
"http://www.w3.org/TR/html4/frameset.dtd">
```

**2．HTML5 的 DTD 定义**

```
<!doctype html>
```

HTML 文档中规定 doctype 是非常重要的，这样浏览器就能了解预期的文档类型。HTML4.01 中的 doctype 需要对 DTD 进行引用，因为 HTML4.01 基于 SGML。而 HTML5 不基于 SGML，因此不需要对 DTD 进行引用，但是需要 doctype 来规范浏览器的行为。

## 2.8 综合实例

以欢度元旦为主题，参照给定的 HTML 代码和图片资源，分别利用 EditPlus、HBuilder 等软件设计 Web 网页，效果如图 2-8 所示。

视频讲解

```
1 <!-- edu_2_8_1.html -->
2 <!doctype html>
3 <html lang="en">
4     <head>
5         <meta charset="UTF-8">
6         <title>标记语法及属性语法应用 </title>
7         <style type="text/css">
8             div{text-align:center;/* 文本居中对齐 */}
9         </style>
10    </head>
11    <body bgcolor="#CDEBE6">
12        <h3 align="center">欢度新年元旦</h3>
13        <hr size="2" color="red" width="100%"/>
14        <p align="left">    元旦（New Year's Day，New
          Year ），指一年开始的第一天，也被称为"新历年"、"阳历年"，在古代指阴历的正
          月初一。1949年9月27日，中国人民政治协商会议第一届全体会议正式确立公历1月1
          日为元旦。元旦是世界上很多国家或地区的法定假日。</p>
15        <div id="" class="">
16            <img src="yundan1.jpg" width="300" height="165" >
17            <img src="yundan2.jpg" width="300" height="165" >
18        </div>
19        <hr size="2" color="red" width="100%"/>
20        <p align="center">Web前端开发技术工作室 Copyright &copy;2017-2020
          </p>
21    </body>
22 </html>
```




图 2-8 综合实例应用

上述代码中第 4～10 行是 HTML 的头部，包含元信息、标题和样式的定义；第 11～21 行是 HTML 的主体，包含标题字、水平分隔线、段落、图层、图像标记定义，其中第

11 行定义主体 body 的背景颜色属性，第 12 行定义标题字 h3 对齐 align 属性（居中对齐），第 13 行、第 19 行定义水平分隔线 hr 的粗细、颜色、宽度等属性，第 14 行定义段落 p 的对齐 align 属性（左对齐），第 15～18 行定义图层，图层内插入 2 个图像 img 标记，分别定义图像的宽度、高度、图片文件的 URL 等属性，第 20 行定义段落，说明版权信息。

# 本 章 小 结

本章主要介绍了 HTML 文件的基本结构。HTML 文档主要包含<html></html>、<head></head>、<body></body>三个标记。

<body></body>之间的内容是 HTML 文档的主体部分，会显示在页面上。body 标记的常用属性有 text、bgcolor、background、link、vlink、alink、topmargin、leftmargin 等。在 head 标记中重点介绍了标题 title 和 meta 元信息标记，其中 meta 标记有两种属性，分别是 name、http-equiv，都是“属性/值”对中的属性，其值由属性 content 给定。

同时介绍了 HTML 标记语法和 HTML 属性语法。HTML 标记分为单个标记和成对标记。成对标记由开始标记和结束标记组成。标记属性必须在开始标记中定义，多个“属性/值”对之间至少空 1 个空格，首个属性与开始标记之间至少空 1 个空格。

HTML4.01 和 XHTML1.0 文档类型有三种：严格型（Strict）、过渡型（Transitional）、框架型（Frameset）。HTML5 文档类型改成简洁型<!doctype html>。

# 练习与实验

## 练习 2

**1. 选择题**

（1）下列标记用于设置页面标题的标记是（　　）。

（A）<caption>　（B）<title>　（C）<html>　（D）<head>

（2）下列标记中能够显示网页内容的标记是（　　）。

（A）<title>　（B）<br>　（C）<html>　（D）<body>

（3）正确表达页面注释格式的是（　　）。

（A）<!--注释 -->　（B）<-- 注释-->

（C）<! 注释>　（D）<! comment>

（4）以下属性中不是 meta 元信息标记的属性是（　　）。

（A）name　（B）color　（C）content　（D）http-equiv

（5）设置 body 显示信息颜色为红色的属性是（　　）。

（A）text　（B）color　（C）bgcolor　（D）background

（6）以下标记不是成对标记的是（　　）。

（A）<html>　（B）<br>　（C）<body>　（D）<head>

**2. 填空题**

（1）HTML 文档通常以______或______作为后缀名，网站的首页文件通常命名为

__________或__________。

（2）HTML 文档是用来描述网页的，一般是由_______和______两部分组成。

（3）HTML 中标记分______标记和______标记两种。部分标记是单个标记，大多数标记是______标记，由_____标记（或______标记）和______标记（或______标记）组成。

（4）HTML4.01 或 XHTML1.0 的文档类型有三种，分别是______、______、______。HTML5 中文档类型定义使用标记正确的写法是___________。

**3．简答题**

（1）简述一个 HTML 文档包应含几个基本标记。并举例说明。

（2）写出 HTML 文件命名规则。

## 实验 2

1．分别使用 EditPlus 或 HBuilder 等软件编写符合以下要求的文档：标题为“求知家园”，在浏览器窗口中显示“欢迎来到我们的求知家园”，完成后的效果如图 2-9 所示。其中网页信息颜色为 blue、背景颜色为#99ffff；水平分隔线粗细为 5、颜色为#ff3333。

2．使用 EditPlus 软件编写符合以下要求的文档：标题为“Google 搜索”，在浏览器窗口中显示“欢迎使用 Google 搜索！”和 Google 图片，完成后的效果如图 2-10 所示。其中网页背景颜色为#ffff33；水平分隔线粗细为 5、颜色为# 0033ff；图片名称为“google.png”。

网页中插入图片的方法格式如下：

```
<img src="google.png" width="275" height="95" border="0" alt="">
```




图 2-9　求知家园页面




图 2-10　Google 搜索页面

# 第 3 章　格式化文本与段落

**本章学习目标**

网页内容排版包括文本格式化、段落格式化和整个页面的排版格式化，这是设计一个网页的基础。文本格式化标记分为字体标记、文字修饰标记。字体标记和文字修饰标记包括对于字体样式的一些特殊修改。段落格式分为段落标记、换行标记、水平分隔线标记等。

通过文本与段落格式化知识的学习，能够掌握页面内容的初步设计，理解并掌握HTML标题字标记、空格及特殊符号的使用。理解格式化标记中的文本修饰标记、计算机输出标记、引用和术语标记的语法及字体 font 标记语法及使用；理解段落与排版标记的语法，学会编写简易的 Web 页面代码。

Web 前端开发工程师应掌握以下内容:

- 掌握标题字（h1 ~ h6）标记语法及属性语法。
- 理解文本格式化标记类型与作用，并学会使用各种样式。
- 学会使用字体 font 标记。
- 学会使用段落与排版标记。
- 学会使用各类格式化标记设计简易的 Web 页面。

## 3.1　Web 页面初步设计

Web 页面设计需要遵循简洁、一致性、有好的对比度的设计原则。简洁是指以满足人们的实际需求为目标，要求简练，准确。一致性是指网站中各个页面使用相同的页边距，页面中的每个元素与整个页面以及站点的色彩和风格保持一致。对比度的目的在于强调突出关键内容，以吸引浏览者，鼓励他们去发掘更深层次的内容。

### 3.1.1　向 Web 页面添加文字信息

在 HTML 文件中，主体内容被包含在<body></body>标记之间，同时 body 标记也有很多自身的属性，例如设置页面背景、设置页面边界等。

**1．基本语法**

```
<body>向这里添加内容</body>
```

**2．语法说明**

body 标记定义文档的主体。

body 标记包含文档的所有内容（例如文本、超链接、图像、表格和列表等）。

一个简单的 HTML 文档必须包含最基本的必备的标记。



视频讲解

【例 3-1-1】文档内容的应用。代码如下所示，页面效果如图 3-1 所示。

```
1  <!--edu_3_1_1.html-->
2  <!doctype html>
3  <html lang="en">
4      <head>
5          <meta charset="UTF-8">
6          <title>简易网页</title>
7      </head>
8      <body>
9          文档的内容... ...
10     </body>
11 </html>
```

**3. 代码解释**

代码中第 4～7 行是 HTML 的头部，包含页面标题；第 8～10 行是 HTML 的主体，第 9 行是向主体中添加的文字信息。

图 3-1　添加文档内容

## 3.1.2　标题字标记

标题字标记由 h1～h6 共六种标记组成。标记中的字母 h 是英语 Heading 简称。作为标题字，h1 标记定义最大的标题字，h6 标记定义最小的标题字。h1 标记到 h6 标记属于块级标记，它们必须在 HTML 中首尾成对出现。浏览器会自动地在标题的前后添加空行。

**1. 基本语法**

```
<h1 align="left|center|right|justify">1号标题文字</h1>
<h2 align="left|center|right|justify">2号标题文字</h2>
<h3 align="left|center|right|justify">3号标题文字</h3>
<h4 align="left|center|right|justify">4号标题文字</h4>
<h5 align="left|center|right|justify">5号标题文字</h5>
<h6 align="left|center|right|justify">6号标题文字</h6>
```

**2. 语法说明**

h 后面的数字越小标题字越大。标题字标记的 align 属性用来定义标题字的对齐方式，对齐方式有四种，分别是 left、center、right、justify。但是一般推荐设计者使用 CSS 样式表来定义对齐方式。

视频讲解

【例 3-1-2】标题字标记的应用。代码如下所示，页面效果如图 3-2 所示。

```
<!-- edu_3_1_2.html -->
<!doctype html>
<html lang="en">
    <head>
        <meta charset="UTF-8">
        <title> 标题字应用 </title>
    </head>
    <body>
        <h1 align="center" >Web前端开发技术</h1>
        <h2 align="left" >Web前端开发技术</h2>
```

```
11         <h3 align="center" >Web前端开发技术</h3>
12         <h4 align="right" >Web前端开发技术</h4>
13         <h5 align="justify" >Web前端开发技术</h5>
14         <h6 align="center" >Web前端开发技术</h6>
15     </body>
16 </html>
```




图 3-2　标题字应用

**3. 代码解释**

代码中第 9 行定义 h1 标题字居中显示；第 10 行定义 h2 标题字左对齐，其余代码相似。

标题文字的大小由它们的重要性决定，等级越高的标题字号越大。在设计时要对各级标题有所规划。

## 3.1.3　添加空格与特殊符号

在 HTML 文件中，添加空格的方式与其他文档添加空格的方式不同，网页中通过代码控制来添加空格，而在其他编辑器中通过键盘空格键来输入空格。

**1. 基本语法**

```
<body>
     &lt;&reg;&times;
</body>
```

**2. 语法说明**

在网页中添加空格使用“ ”，其中“nbsp”是指 Non Breaking Space，空格数量与“ ”个数相同。

在网页中插入特殊字符与插入空格符号的方式相同。特殊符号如表 3-1 所示。

**表 3-1　特殊字符对应的代码**

| 显 示 结 果 | 说　　明 | 符 号 代 码 |
|---|---|---|
| | 显示一个空格 |   |
| < | 小于 | < |
| > | 大于 | > |
| & | &符号 | & |
| " | 双引号 | " |
| © | 版权 | © |
| ® | 注册商标 | ® |
| × | 乘号 | × |
| ÷ | 除号 | ÷ |

在 HTML 文件中特殊字符对应的代码，浏览器解释后会显示对应的特殊符号。

**【例 3-1-3】**插入特殊符号的应用。代码如下所示，页面效果如图 3-3 所示。

```
1 <!-- edu_3_1_3.html -->
2 <!doctype html>
3 <html lang="en">
4     <head>
5         <meta charset="UTF-8">
6         <title>插入特殊符号</title>
7     </head>
8     <body>
9             新浪科技讯 北京时间11月21日凌晨消息，台湾市场研究机构Digitimes Research周二发布报告称，预计2013年全球平板电脑销售量将会达到2.1亿台，超过笔记本的年度销售量。<br>
10        <hr color="blue">
11        <p align="center">版权所有&copy;新浪公司</p>
12    </body>
13 </html>
```

**3. 代码解释**

代码中第 9 行开始插入 4 个空格，用于实现首行缩进 2 个字符，行末插入<br>换行，第 10 行插入蓝色的水平分隔线，第 11 行插入版权特殊符号“©”。

视频讲解

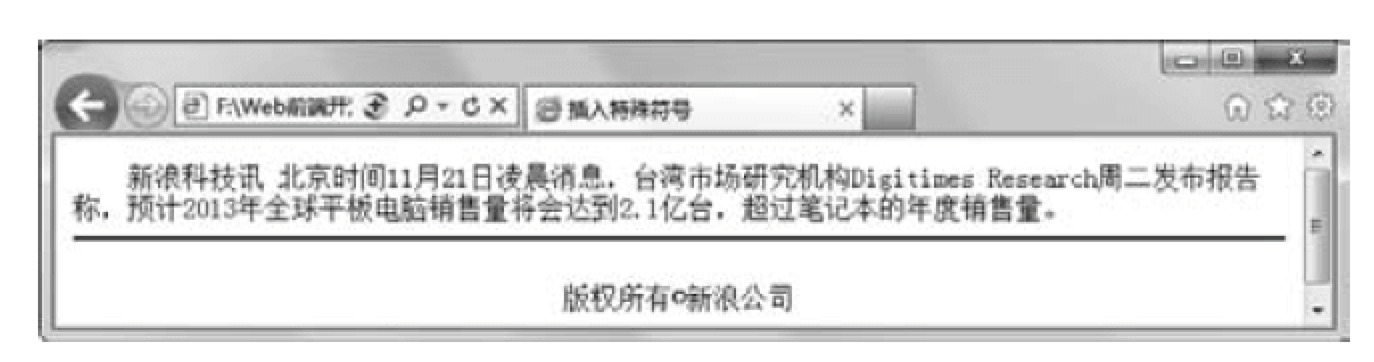



图 3-3 空格与特殊符号的应用

# 3.2 格式化文本标记

HTML 中提供了很多格式化文本的标记，如文字加粗、斜体、下画线、底纹、上下标等。

## 3.2.1 文本修饰标记

文本修饰标记各类浏览器均支持，各类网页开发工具中仍然有这类标记。常见的文本修饰标记如表 3-2 所示。

**表 3-2 常见的文本修饰标记**

| 标 记 | 说 明 |
|---|---|
| <b>软件工程专业！</b> | 定义粗体 |
| <i>软件工程专业！</i> | 定义斜体 |
| <u>软件工程专业！</u> | 定义下画线 |
| <del>软件工程专业！ </del> | 定义删除线 |
| <sup>软件工程专业！</sup> | 定义上标 |
| <sub>软件工程专业！</sub> | 定义下标 |
| <strong>软件工程专业！</strong> | 定义着重文字，与<b></b>效果相同 |
| <em>软件工程专业！</em> | 定义加重语气，与<i></i>效果相同 |
| <small>软件工程专业！</small> | 变小字号 |
| <big>软件工程专业！</big> | 变大字号 |

在 EditPlus V4.0 和 Adobe Dreamweaver CS6 以上版本中，已经使用<em></em>标记来表示强调的文本，替代<i></i>斜体标记；使用<strong></strong>标记来表示重要文本，替代<b></b>粗体标记。但<b></b>标记和<i></i>标记也还在使用。

### 3.2.2 计算机输出标记

常用的计算机输出标记如表 3-3 所示。

**表 3-3 常用的计算机输出标记**

| 标　　记 | 说　　明 |
|---|---|
| <code></code> | 定义计算机代码 |
| <kbd></kbd> | 定义键盘码 |
| <samp></samp> | 定义计算机代码样本 |
| <tt></tt> | 定义打字机代码 |
| <var></var> | 定义变量 |
| <pre></pre> | 定义预格式文本 |

### 3.2.3 引用和术语标记

常用的引用和术语标记如表 3-4 所示。

**表 3-4 常用的引用和术语标记**

| 标　　记 | 主 要 用 途 |
|---|---|
| <abbr>etc.</abbr> | 定义缩写 |
| <address>江苏南京市</address> | 定义地址 |
| <blockquote>长的引用</blockquote> | 定义长的引用 |
| <cite>引用、引证</cite> | 定义引用、引证 |
| <q>引用短语</q> | 定义短的引用语，IE 看不到引号，其余可以 |
| <dfn>定义项目</dfn> | 定义一个定义项目 |

**【例 3-2-1】**文本修饰标记的应用。代码如下所示，页面效果如图 3-4 所示。

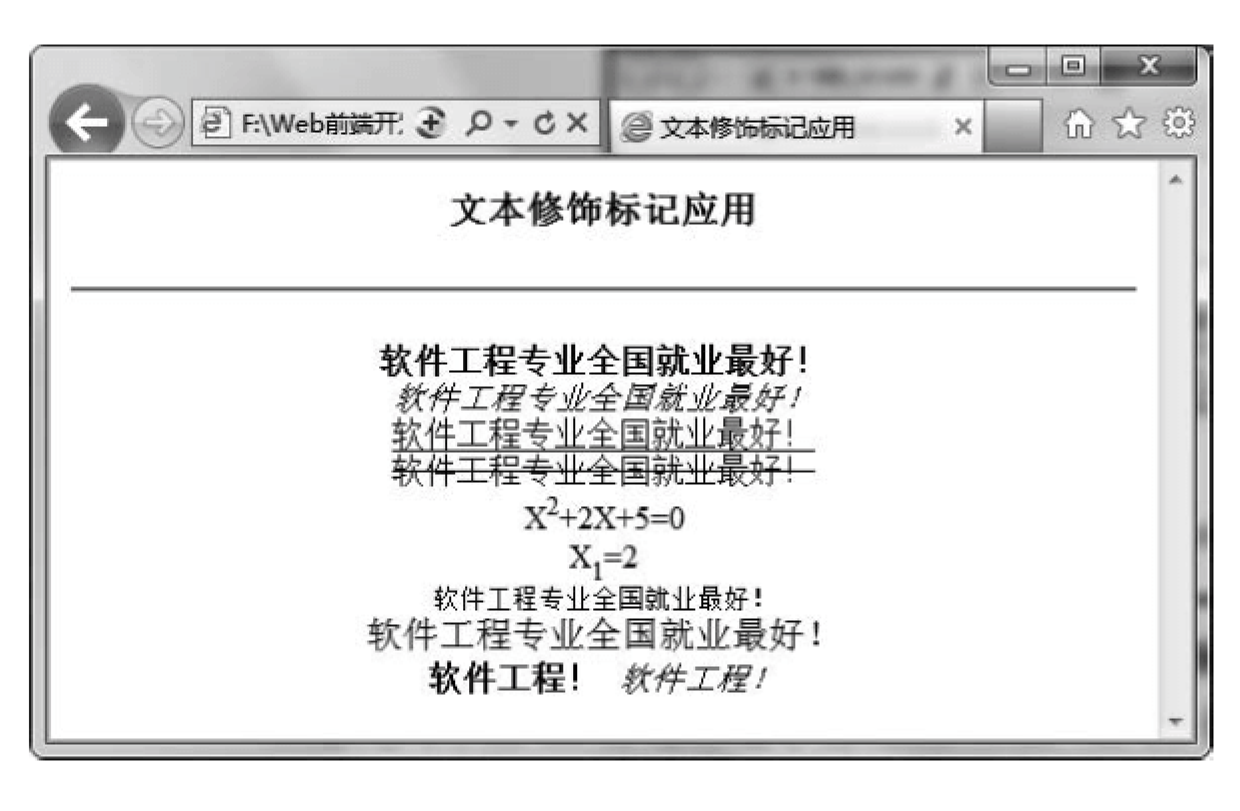



图 3-4 文本修饰标记应用



视频讲解

```
<!-- edu_3_2_1.html -->
<!doctype html>
<html lang="en">
```

```
4  <head>
5   <meta charset="UTF-8">
6   <title>文本修饰标记应用</title>
7   <style type="text/css">
8       *{text-align:center;/* 所有标记的内容居中显示 */}
9   </style>
10 </head>
11 <body >
12     <h3 align="center">文本修饰标记应用</h3>
13     <hr size="2" color="red">
14     <comment>修饰标记应用</comment>
15     <b>软件工程专业全国就业最好! </b><br>
16     <i>软件工程专业全国就业最好! </i><br>
17     <u>软件工程专业全国就业最好! </u><br>
18     <del>软件工程专业全国就业最好! </del><br>
19      X<sup>2</sup>+2X+5=0<br>
20      X<sub>1</sub>=2<br>
21     <small>软件工程专业全国就业最好! </small><br>
22     <big>软件工程专业全国就业最好! </big><br>
23     <strong>软件工程! </strong>
24     <em>软件工程! </em>
25   </body>
26 </html>
```

上述代码中第 12 行是标题字标记的应用；第 14 行注释标记应用；第 15～24 行定义不同的文本修饰标记。

计算机输出标记和引用和术语标记在 3.3 节中另行举例，此处省略。

### 3.2.4 字体 font 标记

在不指定任何样式的情况下，IE 浏览器会把字体显示为 3 号、黑色、宋体。因此设计网页时，根据需要更改不同段落的字体。HTML5 中可以使用 CSS 中的字体属性替代。

font 标记规定文本的字体系列、字体尺寸、字体颜色，所有浏览器均支持 font 标记。

**1. 基本语法**

```
<font face=""  size=""  color="" >…</font>
```

**2. 属性说明**

font 标记的属性、值及其说明如表 3-5 所示。

**表 3-5 font 标记的属性、值及其说明**

| 属　性 | 值 | 说　明 |
|---|---|---|
| size | +1～+7，1～7，-1～-7 | 正数字越大字号越大，负数字越大字号越小。“+”表示字号比原来的字号大一些，“-”表示字号比原来的字号小一些 |
| color | rgb（r,g,b）rgb（r%,g%,b%）<br>#rrggbb 或#rgb<br>colorname | 规定文本的颜色。可以使用 rgb 函数、十六进制数、颜色英文名称来表达 |
| face | 字体 1，字体 2，…，字体 n | face 属性可以有多个值，用逗号分隔。字体使用方式为从左向右依次选用。如果前面的字体不存在，则使用后一个字体。若都不存在，则默认使用“宋体” |

【例 3-2-2】网页字体样式的应用。代码如下所示，页面效果如图 3-5 所示。

视频讲解

```
1 <!-- edu_3_2_2.html -->
2 <!doctype html>
3 <html lang="en">
4  <head>
5   <meta charset="UTF-8">
6         <title>文字样式</title>
7     </head>
8     <body>
9         <strong>文字样式为黑体、颜色#000fff、大小从-1～-7:</strong>
10        <font face="黑体" size="-1" color="#000fff">-1字</font>
11        <font face="黑体" size="-3" color="#000fff">-3字</font>
12        <font face="黑体" size="-5" color="#000fff">-5字</font>
13        <font face="黑体" size="-7" color="#000fff">-7字</font><br>
14        <strong>文字样式为宋体、颜色#ff0066、大小从1～7:</strong>
15        <font face="宋体" size="2" color="#ff0066">2字</font>
16        <font face="宋体" size="4" color="#ff0066">4字</font>
17        <font face="宋体" size="6" color="#ff0066">6字</font>    <br>
18        <strong>文字样式为隶书、颜色#ff0066、大小从+1～+7: </strong>
19        <font face="黑体" size="+1" color="#ff0066">1字</font>
20        <font face="黑体" size="+3" color="#ff0066">3字</font>
21        <font face="黑体" size="+5" color="#ff0066">5字</font>
22        <font face="黑体" size="+7" color="#ff0066">7字</font>
23    </body>
24 </html>
```

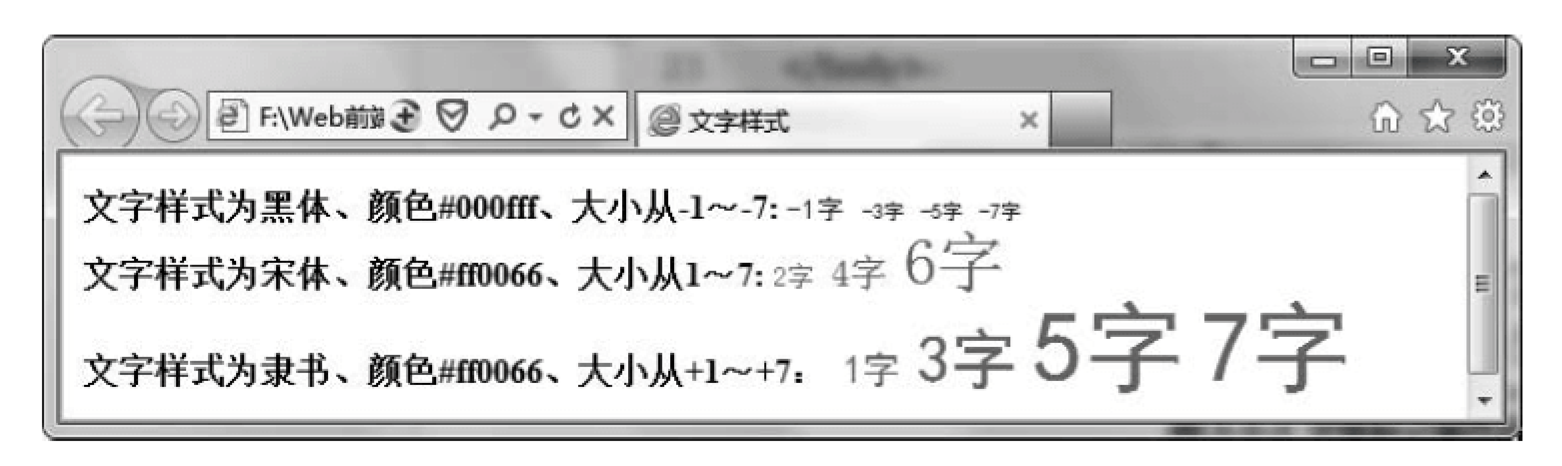



图 3-5　字体标记属性应用

**3. 代码解释**

代码中第 10～13 行设置字体为“黑体、颜色为#000fff、大小从-1～-7”；第 15～17 行设置字体为“宋体、颜色为#ff0066、大小从 1～7”；第 19～22 行设置字体为“黑体、颜色为#ff0066、大小为+1～+7”。

# 3.3　段落与排版标记

网页的外观是否美观，很大程度上取决于其排版。在页面中出现大段的文字，通常采用分段进行规划，对换行也有极其严格的划分。本节从段落的细节设置入手，利用段落与排版标记自如地处理大段的文字。

## 3.3.1　段落 p 标记

在 HTML 文档中，合理使用段落会使文字的显示更加美观，表达更加清晰。段落 p 标

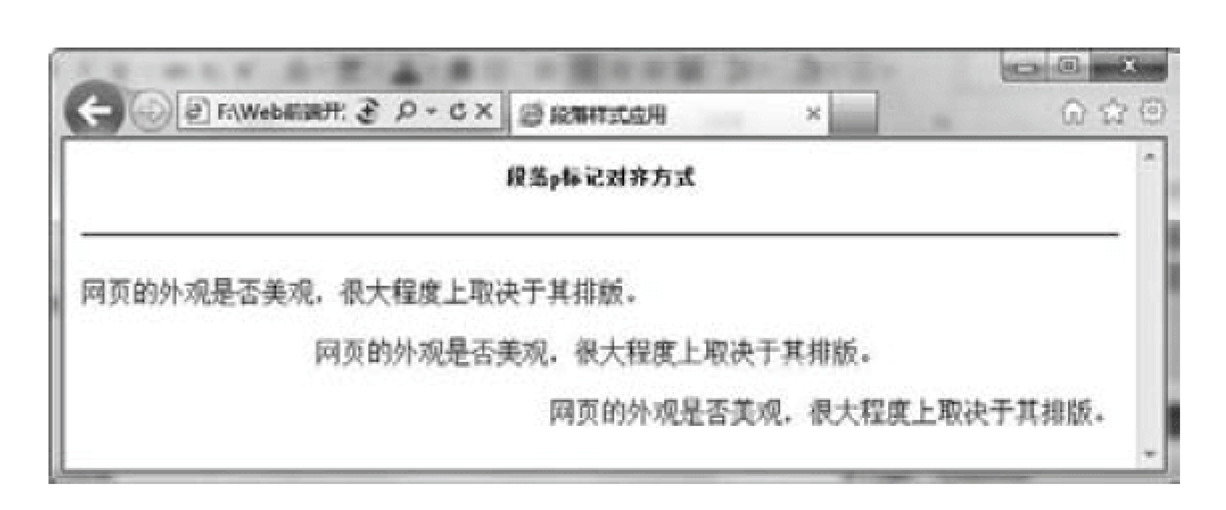
记用来开始一个段落，它是一个块级标记，该标记中不能再包含其他的任何块级标记。

**1. 基本语法**

```
<p align="left|center|right|justify">段落正文内容</p>
```

p 标记会自动在其前后创建一些空白。浏览器会自动添加这些空间。段落 p 标记的 align 属性有四个属性值，分别表示左对齐、居中对齐、右对齐、两端对齐。

【例 3-3-1】网页段落样式的应用。代码如下所示，页面效果如图 3-6 所示。





视频讲解

图 3-6　段落样式应用

```
 1 <!-- edu_3_3_1.html -->
 2 <!doctype html>
 3 <html lang="en">
 4     <head>
 5         <meta charset="UTF-8">
 6         <title>段落样式应用</title>
 7     </head>
 8     <body>
 9         <h5 align="center">段落p标记对齐方式</h5>
10         <hr color="blue">
11         <p align="left">网页的外观是否美观，很大程度上取决于其排版。</p>
12         <p align="center">网页的外观是否美观，很大程度上取决于其排版。</p>
13         <p align="right">网页的外观是否美观，很大程度上取决于其排版。</p>
14     </body>
15 </html>
```

**2．代码解释**

代码中第 4～7 行是 HTML 的头部，包含页面标题；第 8～14 行是 HTML 的主体，包含多种段落样式，其中第 11 行为左对齐，第 12 行为居中对齐，第 13 行为右对齐格式。

## 3.3.2　换行 br 标记

在 HTML 文件中，插入换行标记<br>的作用和普通文档插入回车的作用一样，都表示强制性换行。

基本语法

```
<br>或<br/>
```

在 HTML 文档中，换行 br 标记属于单标志，表示插入换行符。

## 3.3.3　水平分隔线 hr 标记

水平分隔线标记用一条线将页面区域按照功能用途进行分隔。hr 标记是单个标记。

**1. 基本语法**

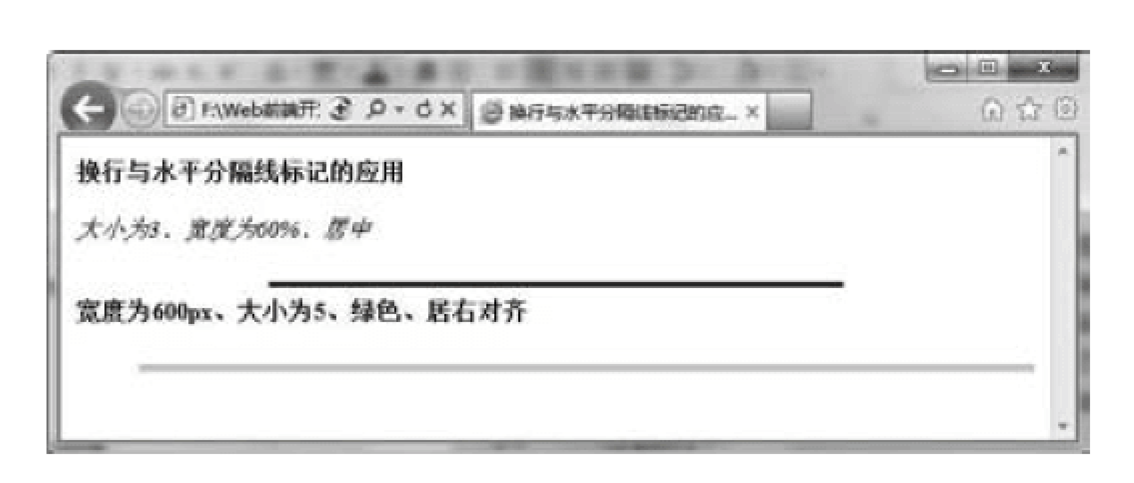
```
<hr width="" size="" color="" align="" noshade>
```

水平分隔线 hr 标记的属性、值及其说明如表 3-6 所示。

**表 3-6　<hr>标记的属性、值及其说明**

| 属　性 | 值 | 说　明 |
| --- | --- | --- |
| width | 像素（px）或百分比 | 设置水平线宽度 |
| size | 整数，单位 px | 设置水平线高度 |
| color | rgb 函数、十六进制数，颜色英文名称 | 设置水平线颜色 |
| align | left\|center\|right | 设置水平线对齐方式 |

**【例 3-3-2】**换行与水平分隔线标记的应用。代码如下所示，页面效果如图 3-7 所示。






视频讲解

图 3-7　插入水平分隔线

```
1 <!-- edu_3_3_2.html -->
2 <!doctype html>
3 <html lang="en">
4     <head>
5         <meta charset="UTF-8">
6         <title>换行与水平分隔线标记的应用</title>
7     </head>
8     <body>
9         <h4>换行与水平分隔线标记的应用</h4>
10        <p><em>大小为3、宽度为60%、居中</em></p>
11        <hr size="3" width="60%" color="#330099" noshade>
12        <strong>宽度为600px、大小为5、绿色、居右对齐</strong><br><br>
13        <hr width="600px" size="5" color="#00ee99" align="right">
14    </body>
15 </html>
```

**2. 代码解释**

代码中第 11 行插入 1 条“大小为 3、宽度为 60%、居中”的水平分隔线；第 13 行插入 1 条“宽度为 600px、大小为 5、居右对齐”的水平分隔线。

## 3.3.4　拼音/音标注释 ruby 标记和 rt/rp 标记

ruby 标记定义 ruby 注释（中文注音或字符）。ruby 标记与 rt 标记一同使用。ruby 标记由一个或多个字符（需要一个解释/发音）和一个提供该信息的 rt 标记组成，还包括可选的 rp 标记，定义当浏览器不支持 ruby 标记时显示的内容。效果如图 3-8 所示。

ruby 标记用它将需要注释或注音标的文字内容包围住。

rt 标记这里面放置音标或注释，这个标记要跟在需要注释的文本后边。

rp 标记是防备那些不支持 ruby 标记的浏览器，主要用来放置括号。对于支持这个标记的浏览器，rp 标记的 CSS 样式是{display:none;}，也就是不可见。

基本语法

```
<ruby>
   中<rt><rp>(</rp>zhong<rp>)</rp></rt>
   国<rt><rp>(</rp>guo<rp>)</rp></rt>
</ruby>
```

zhongg u o

中 国

图 3-8　ruby 标记的应用

## 3.3.5　段落缩进 blockquote 标记

段落缩进 blockquote 标记是块级标记，常称为块引用标记。该标记引用的内容能够向右缩进 5 个英文字符的位置，并在其内容的周围增加外边距。

**1. 基本语法**

```
<blockquote>引用的内容</blockquote>
```

**【例 3-3-3】**拼音/音标注释标记与块引用标记的应用。代码如下所示，页面效果如图 3-9 所示。

视频讲解

```
<!-- edu_3_3_3.html -->
<!doctype html>
<html lang="en">
 <head>
  <meta charset="UTF-8">
        <title>注释与块引用标记的应用</title>
        <style type="text/css">
           ruby{font-size:58px;font-family:黑体;text-align:center;}
        </style>
    </head>
    <body>
     <h5>注释ruby标记-标注读音</h5>
     <p align="center"><ruby>
             智<rt><rp>(</rp>zhì<rp>)</rp></rt>
             慧<rt><rp>(</rp>huì<rp>)</rp></rt>
             地<rt><rp>(</rp>dì<rp>)</rp></rt>
             球<rt><rp>(</rp>qiú<rp>)</rp></rt>
     </ruby></p>
      <h5>段落缩进标记的应用</h5>
        <hr color="green">
        <p>这行文字没有缩进</p>
        <blockquote>这行文字行首缩进5个字符位置</blockquote>
        <blockquote><blockquote>这行文字行首缩进10个字符位置</blockquote>
        </blockquote>
    </body>
</html>
```




图 3-9　设置段落缩进

**2. 代码解释**

代码中第 13～18 行设置 ruby 标记标注汉语拼音。第 21 行，此行文字没有设置块引用，所以没有缩进；第 22 行设置块引用，所以此行文字行首缩进 5 个字符位置；第 23 行嵌套使用 2 个块引用标记，此行行首向右缩进 10 个字符的位置。

## 3.3.6　预格式化 pre 标记

在 HTML 中利用成对的<pre></pre>标记对网页中的文字段落进行预格式化，浏览器会完整保留设计者在源文件中所定义的格式，包括各种空格、缩进以及其他特殊格式。

**1. 基本语法**

```
<pre>预格式化文本 </pre >
```

【例 3-3-4】预格式化的应用。代码如下所示，页面效果如图 3-10 所示。



视频讲解

```
<!-- edu_3_3_4.html -->
<!doctype html>
<html lang="en">
   <head>
      <meta charset="UTF-8">
      <title>预格式化</title>
   </head>
   <body>
     <h1><pre>
                  春 晓

                  孟浩然
                春眠不觉晓，
                                    处处闻啼鸟。
                夜来风雨声，
                                    花落知多少。
      </pre></h1>
   </body>
</html>
```

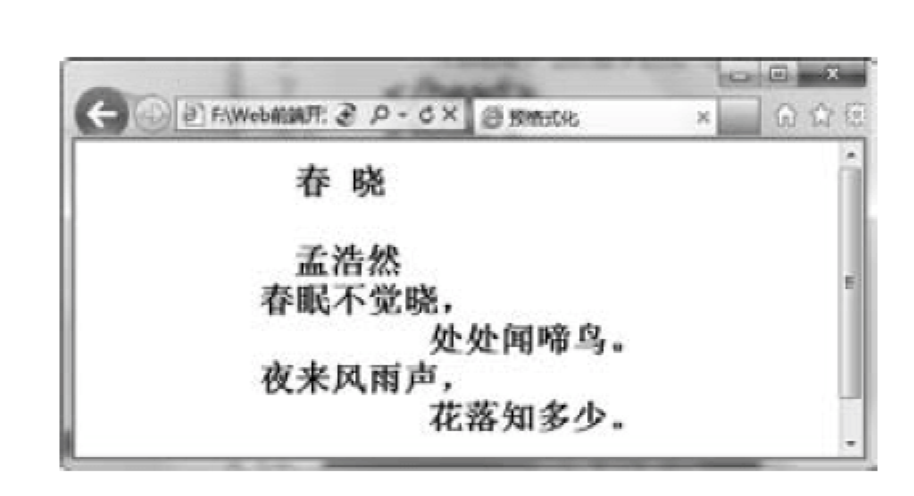



图 3-10　预格式化

**2. 代码解释**

代码中第 4～7 行是 HTML 的头部，包含元信息、页面标题；第 8～18 行是 HTML 的主体，其中第 9～17 行对文字段落进行预格式化。

## 3.4　综 合 实 例

视频讲解

以“教育信息化‘十三五’规划报告”为主题，参照给定的 HTML 代码，完成 Web 网页的设计，页面效果如图 3-11 所示。

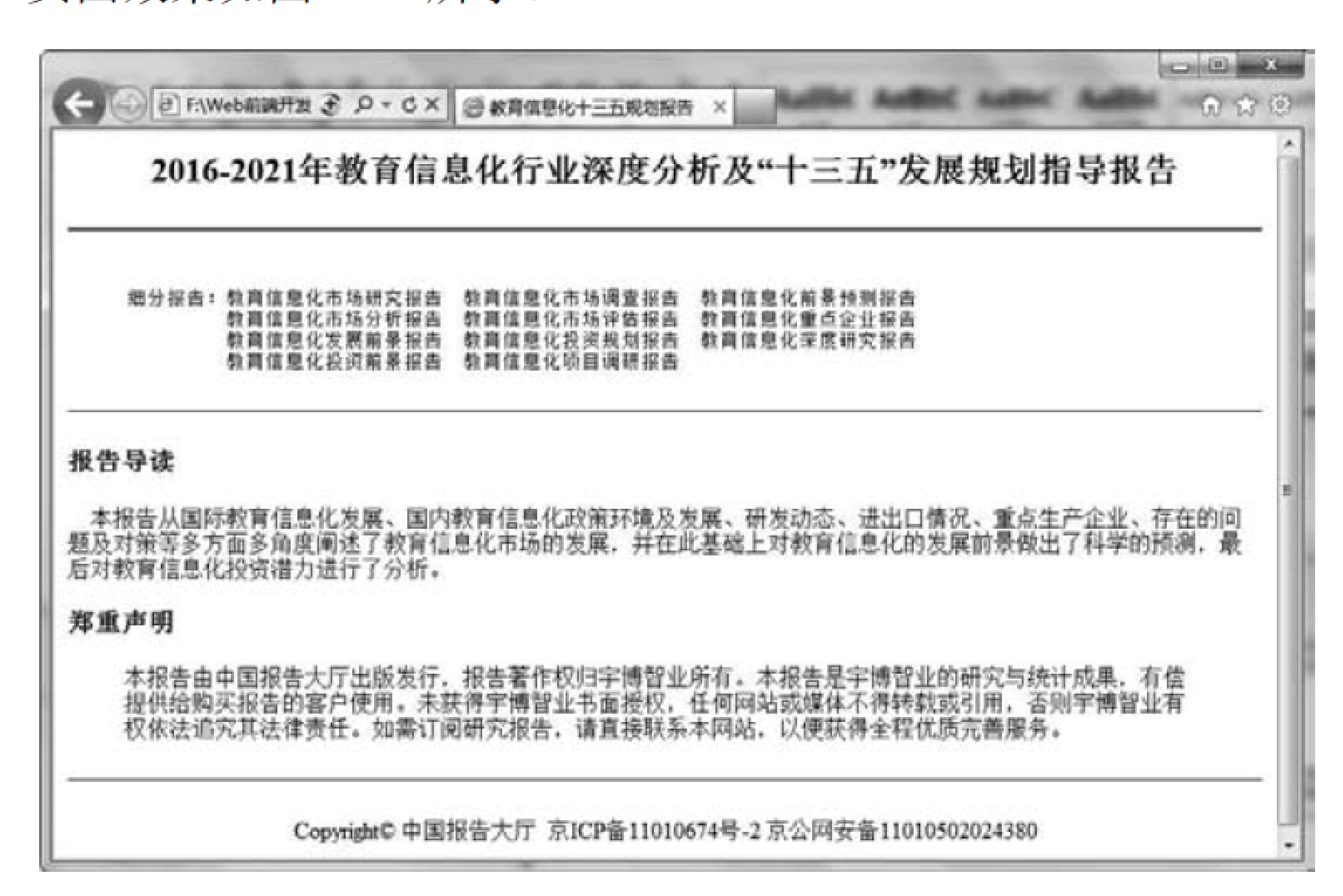



图 3-11　教育信息化“十三五”规划报告页面

```
<!-- edu_3_4_1.html -->
<!doctype html>
<html lang="en">
    <head>
        <meta charset="UTF-8">
        <title>教育信息化"十三五"规划报告</title>
    </head>
    <body>
        <h2 align="center">2016-2021年教育信息化行业深度分析及“十三五”发展规划指导报告</h2>
        <hr width="100%" size="3" color="red">
        <pre>
                细分报告：教育信息化市场研究报告　教育信息化市场调查报告　教育信息化前景预测报告
                                教育信息化市场分析报告　教育信息化市场评估报告　教育信息化重点企业报告
                                教育信息化发展前景报告　教育信息化投资规划报告　教育信息化深度研究报告
                                教育信息化投资前景报告　教育信息化项目调研报告
        </pre>
        <hr width="100%" size="1" color="#000fff">
        <h4>报告导读</h4>
        <p>    本报告从国际教育信息化发展、国内教育信息化政策环境及发展、研发动态、进出口情况、重点生产企业、存在的问题及对策等多方面
```

```
         多角度阐述了教育信息化市场的发展，并在此基础上对教育信息化的发展前景做出了科
         学的预测，最后对教育信息化投资潜力进行了分析。</p>
20       <h4>郑重声明</h4>
21       <p><blockquote>本报告由中国报告大厅出版发行，报告著作权归宇博智业所有。
         本报告是宇博智业的研究与统计成果，有偿提供给购买报告的客户使用。未获得宇博
         智业书面授权，任何网站或媒体不得转载或引用，否则宇博智业有权依法追究其法律
         责任。如需订阅研究报告，请直接联系本网站，以便获得全程优质完善服务。
         </blockquote></p>
22       <hr width="100%" size="1" color="#000fff">
23       <p align="center">Copyright&copy; 中国报告大厅 京ICP备11010674号-2
         京公网安备11010502024380</p>
24    </body>
25 </html>
```

上述代码中第 4～7 行是 HTML 的头部；第 8～24 行是 HTML 的主体，其中第 10 行、第 17 行定义 2 条水平分隔线；第 11～16 行应用预格式化标记；第 18 行和第 20 行应用标题字 h4 标记；第 19 行和第 21 行定义 2 个段落，分别应用空格和段落缩进标记；第 23 行应用段落居中和特殊符号。

# 本 章 小 结

本章主要介绍了格式化文字与段落的各种标记，包括标题字标记、字体标记、文本修饰标记以及段落相关的标记。<h1>～<h6>是标题字标记，通过 align 属性设置标题字的对齐方式。空格与特殊字符都需要通过代码控制来添加。字体标记主要通过 font 标记的属性改变字体、颜色、大小。文本修饰标记主要是对文本进行一些特殊的修饰。

段落与排版标记会使网页文字显得更加清晰，介绍了段落 p 标记、换行 br 标记、水平分隔线 hr 标记、注释 ruby 标记、段落缩进 blockquote 标记的使用方法。

在网页设计中，对网页的文字进行必要的布局并添加页面效果，从而使网页更加美观和丰富，要合理地使用本节介绍到的各种文字和段落标记。

# 练习与实验

## 练习 3

**1. 选择题**

（1）下列不是字体标记的属性的是（　　）。

（A）align　　（B）size　　（C）color　　（D）face

（2）关于标题字标记对齐方式，标记属性取值不正确的是（　　）。

（A）居中对齐：<h1 align="middle">…</h1>

（B）居右对齐：<h2 align="right">…</h2>

（C）居左对齐：<h4 align="left">…</h4>

（D）两端对齐：<h6 align="justify">…</h6>

（3）下列选项中表示字体标记的是（　　）。

（A）<boby></body>　（B）<font> </font>　（C）<br>　（D）<p></p>

（4）下列选项中表示段落标记的是（　　）。

（A）<html>< / html>　（B）<boby></body>

（C）<p></p>　（D）<pre>< / pre>

（5）在 HTML 中，<h3>< / h3>是（　　）标记。

（A）标题字　（B）预格式化　（C）换行　（D）随意显示信息

（6）下列标记中，设置页面标题的标记是（　　）。

（A）<title></title>　（B）<caption></caption>

（C）<head> </head>　（D）<html></html>

（7）下列标记中表示单个标记的是（　　）。

（A）body 标记　（B）br 标记

（C）html 标记　（D）title 标记

（8）<title></title>标记是放在（　　）标记内。

（A）<pre>　</pre>　（B）<head>　</head>

（C）<body>　</body>　（D）</head> <body>

（9）下列选项中表示版权符号的是（　　）。

（A）<　（B）>　（C）®　（D）©

（10）HTML 中<hr>的作用是（　　）。

（A）插入一条水平分隔线　（B）换行

（C）插入一个空格　（D）加粗字体

**2. 填空题**

（1）HTML 网页文件的主体标记是______，设置页面标题的标记是______。

（2）一个 HTML 文档的开始标记是______；结束标记是______。

（3）设置文档标题以及其他不在 Web 网页上显示的信息的开始标记是______；结束标记是______。

（4）网页中可显示的信息是包含在以______为开始标记，以______为结束标记之间。

（5）网页标题会显示在浏览器的标题栏中，则网页标题可使用______标记来定义。

（6）与标记<b></b>功能相同的标记是________；与标记<i></i>功能相同的标记是__________。

（7）__________标记是由一个或多个需要解释/发音的字符和一个提供该信息的_________标记组成，还包括可选的_____________标记，定义当浏览器不支持 ruby 标记时显示的内容。

**3. 简答题**

（1）简述格式化文本标记分几类，并举例说明。

（2）简述有哪些段落与排版标记及其作用。

## 实验 3

**1.** 编写代码实现如图 3-12 所示的页面效果。设计要求：页面上方水平分隔线粗细为 1px、颜色为#000fff，页面下方水平分隔线粗细为 1px、颜色为#00ffff。




图 3-12　自荐信页面

**2.** 按如下要求设计 Web 页面，如图 3-13 所示。要求如下：

（1）3 号标题字设置标题“数学方程式”，样式采用 style 标记定义，格式为字体大小 24px、颜色红色、文本居中对齐；

（2）一条宽度为 80%、大小为 2px、颜色为蓝色的水平线；

（3）方程式 1：$2x^2+3x=9$;

（4）方程式 2：$x_1+x_2=10$;

（5）在头部插入样式标记，定义如下：

```
<style type="text/css">
    h3{font-size:24px;color:red;text-align:center;}
</style>
```

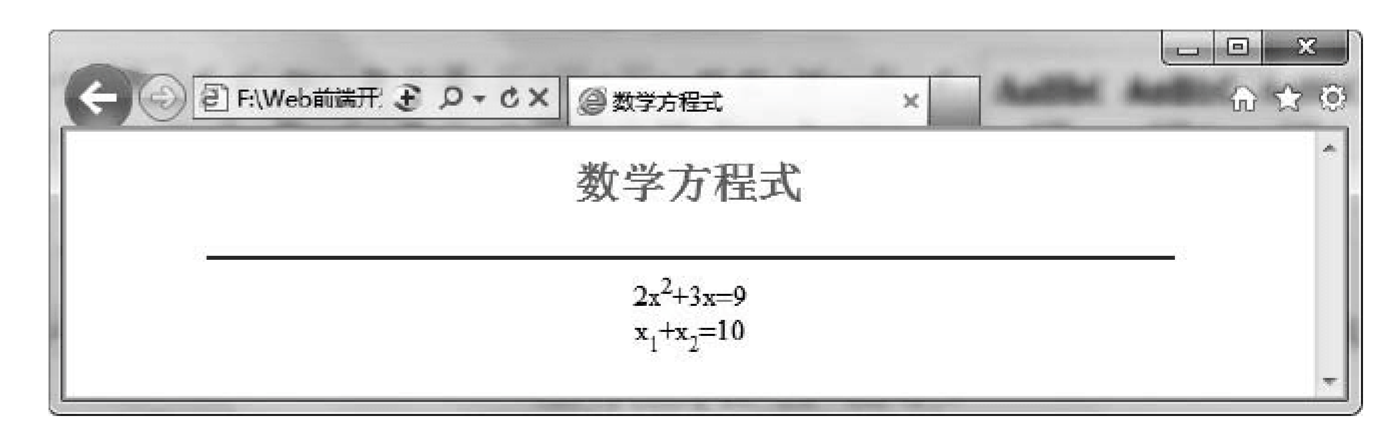



图 3-13　标题字标记及文本标记的应用

# 第 4 章　列　表

本章学习目标

大型 IT 网站如网易、搜狐、新浪等的首页导航栏目均采用列表方式来显示信息。通过列表知识的学习，能够了解列表的类型，掌握无序列表、有序列表、定义列表的作用及使用方法；学会使用不同列表类型及嵌套列表来解决网页设计中遇到的一些实际问题。

Web 前端开发工程师应掌握以下内容：

- 了解列表的类型。
- 掌握无序列表、有序列表、定义列表标记语法及属性语法。
- 学会使用无序、有序及定义列表设计 Web 网页。
- 学会使用嵌套列表设计小型网站首页。

## 4.1　列 表 概 述

列表能对网页中的相关信息进行合理的布局，将项目有序或无序地罗列在一起，便于用户浏览和操作。列表分为无序列表、有序列表、定义列表、菜单列表和目录列表共五种。但常用列表有三种，分别是无序列表、有序列表、定义列表。列表类型如表 4-1 所示。

表 4-1　列表类型与标记符号

| 列 表 类 型 | 标 记 符 号 | 备　　注 |
|---|---|---|
| 无序列表 | <ul>…</ul> | 常用 |
| 菜单列表 | <menu>…</menu> | 不常用 |
| 目录列表 | <dir>…</dir> | 不常用 |
| 有序列表 | <ol>…</ol> | 常用 |
| 定义列表 | <dl>…</dl> | 常用 |

## 4.2　无 序 列 表

无序列表 ul（Unordered List）标记是成对标记，<ul>是开始标记，</ul>是结束标记，两者之间插入若干个列表项 li（List Items）标记，完成无序列表的插入。

**1. 基本语法**

```
<ul type="">
    <li type="">项目名称</li>
    <li type="">项目名称</li>
    …
</ul>
```

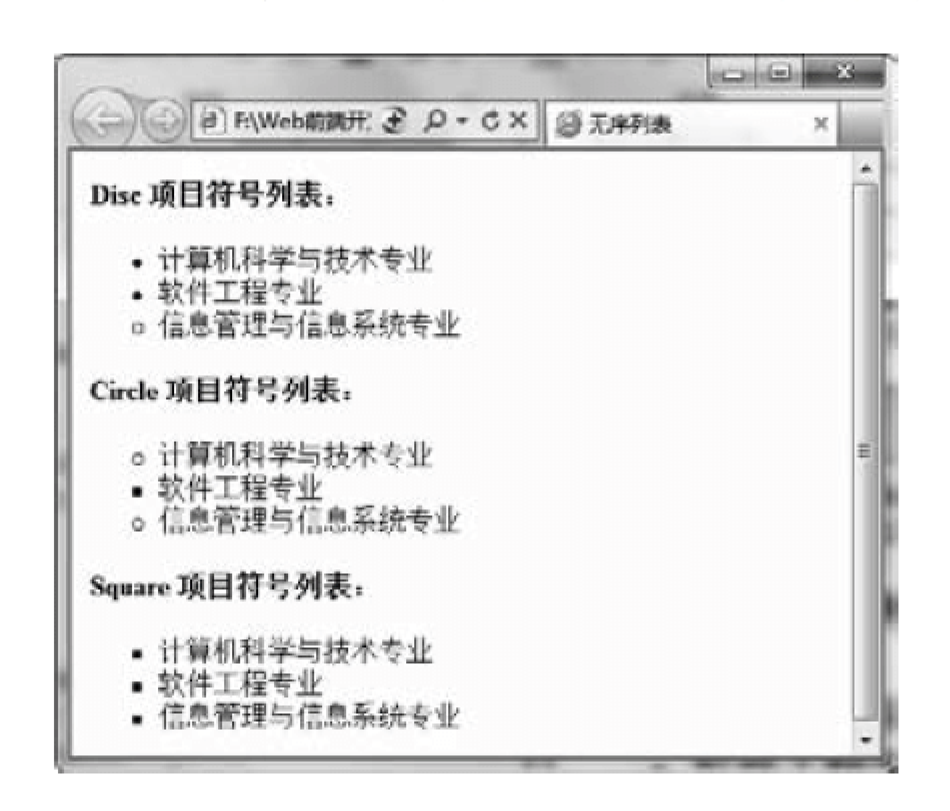
**2. 语法说明**

ul 标记的 type 属性有三个值，如表 4-2 所示。列表项 li 标记的 type 属性取值与 ul 标记相同。设置 ul 标记的 type 属性会使其所包含的列表项按统一风格显示，设置其中某一列表项的 type 属性值时只会影响它自身的显示风格，其他列表项按原样显示。

表 4-2　无序列表标记的 type 属性及其说明

| 属　性　值 | 说　　明 |
|---|---|
| disc | 实心圆形● |
| circle | 空心圆形○ |
| square | 实心正方形■ |

【例 4-2-1】无序列表的应用。代码如下所示，页面效果如图 4-1 所示。






视频讲解

图 4-1　无序列表的应用

```
<!-- edu_4_2_1.html -->
<!doctype html>
<html lang="en">
    <head>
        <meta charset="UTF-8">
        <title>无序列表</title>
    </head>
    <body>
        <h4>Disc 项目符号列表：</h4>
        <ul type="disc">
            <li>计算机科学与技术专业</li>
            <li>软件工程专业</li>
            <li type="circle">信息管理与信息系统专业</li>
        </ul>
        <h4>Circle 项目符号列表：</h4>
        <ul type="circle">
            <li>计算机科学与技术专业</li>
            <li type="square">软件工程专业</li>
            <li>信息管理与信息系统专业</li>
        </ul>
        <h4>Square 项目符号列表：</h4>
        <ul type="square">
            <li>计算机科学与技术专业</li>
            <li>软件工程专业</li>
            <li>信息管理与信息系统专业</li>
        </ul>
```

```
27     </body>
28 </html>
```

**3. 代码解释**

代码中第 10～14 行列表符号为实心圆形，除第 13 行定义了列表项的 type 属性值为"circle"，所以此项前面显示空心圆；第 16～20 行列表符号为空心圆形，除第 18 行定义了列表项的 type 属性值为"square"，所以此项前面显示实心正方形；第 22～26 行列表符号为实心正方形。通过设置 type 属性值来改变列表项前面的符号。

## 4.3 有 序 列 表

有序列表 ol（Ordered List）标记是成对标记，以<ol>为起始标记，以</ol>为结束标记，在其间使用<li></li>标记完成有序列表项目的插入。

**1. 基本语法**

```
<ol type="" start="">
    <li type=""  value="n">项目名称</li>
    <li type=""  value="n">项目名称</li>
    …
</ol>
```

在<ol>、</ol>标记之间必须使用<li></li>标记来添加列表项值。

**2. 属性说明**

1）列表 ol 标记的属性

- type：列表项前面的编号，编号是有序的，有五种不同类型。
- start：定义有序列表起始编号，默认值为 1。设置其为非 1 时，列表项前编号起始位置会发生改变，如 start="5"，当 type="1"时，表示从第 5 个开始编号；当 type="A"，表示从 E 开始编号，以此类推。

2）列表项 li 标记的属性

- type：只影响当前列表项前面编号类型，后续列表项前面编号类型依旧遵循 ol 标记的 type 属性的取值。
- value：改变当前列表项前编号的值，并影响其后所有列表项编号的值。

有序列表 ol 标记的属性、值及说明如表 4-3 所示。

**表 4-3　有序列表的属性、值及说明**

| 属　性 | 值 | 说　明 |
|---|---|---|
| type | 1 | 定义有序列表中列表项前面的编号为数字列表 |
| | A | 定义有序列表中列表项前面的编号为大写字母列表 |
| | a | 定义有序列表中列表项前面的编号为小写字母列表 |
| | I | 定义有序列表中列表项前面的编号为大写罗马字母列表 |
| | i | 定义有序列表中列表项前面的编号为小写罗马字母列表 |
| start | 数值 | 有序列表中列表项的起始数字 |

【例 4-3-1】有序列表的应用。代码如下所示，页面效果如图 4-2 所示。




```
1 <!-- edu_4_3_1.html -->
2 <!doctype html>
3 <html lang="en">
4     <head>
5         <meta charset="UTF-8">
6         <title>有序列表</title>
7     </head>
8     <body>
9         <h4>1数字编号：</h4>
10        <ol>
11            <li>计算机科学与技术专业</li>
12            <li>软件工程专业</li>
13            <li>信息管理与信息系统专业</li>
14            <li>电子信息工程专业</li>
15        </ol>
16        <h4>A字母编号：</h4>
17        <ol type="A">
18            <li>计算机科学与技术专业</li>
19            <li>软件工程专业</li>
20            <li>信息管理与信息系统专业</li>
21            <li>电子信息工程专业</li>
22        </ol>
23        <h4>aI混合编号：</h4>
24        <ol type="a">
25            <li>计算机科学与技术专业</li>
26            <li type="I" value="5">软件工程专业</li>
27            <li>信息管理与信息系统专业</li>
28            <li>电子信息工程专业</li>
29            <li>电子科学与技术专业</li>
30            <li>物联网工程专业</li>
31        </ol>
32    </body>
33 </html>
```

改变列表项的类型和编号的属性值会影响本身的编号类型和后面列表项编号的顺序

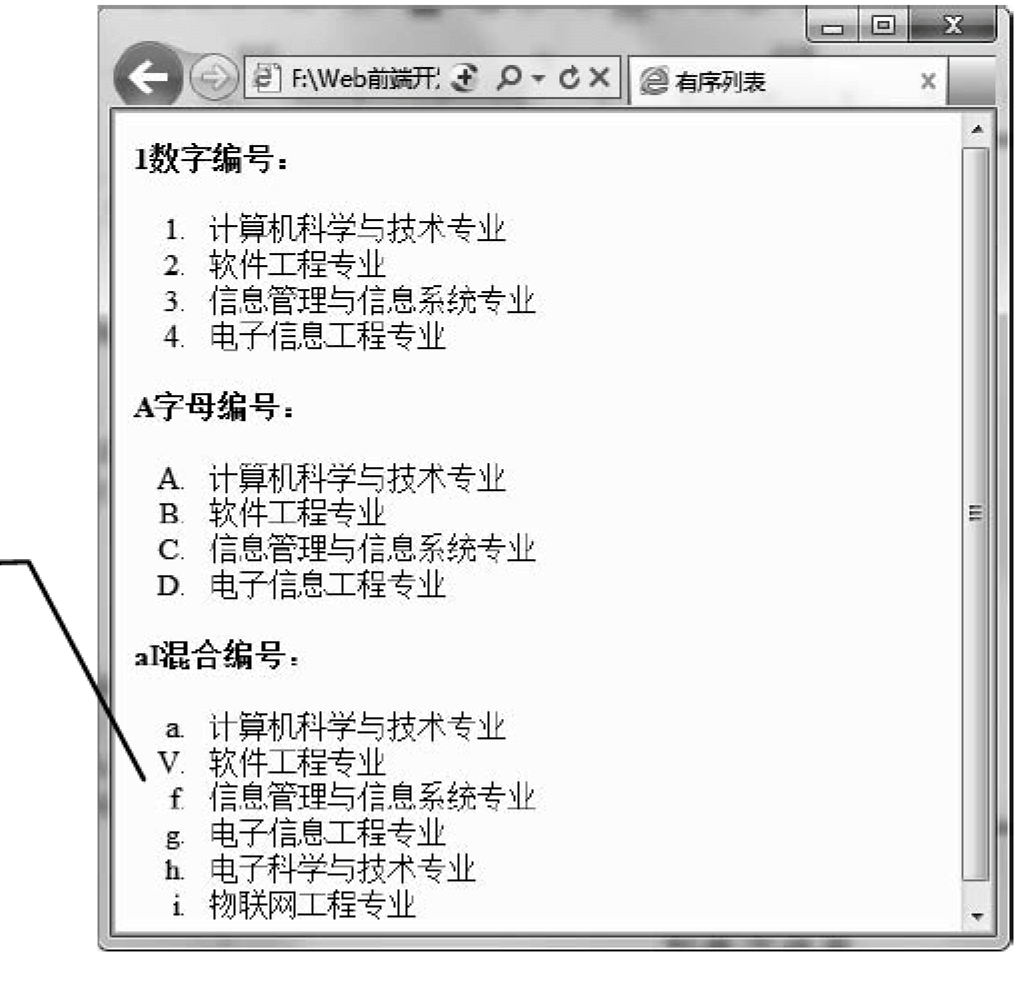



图 4-2　有序列表的应用

**3. 代码解释**

代码中第 10～15 行实现数字编号的有序列表；第 17～22 行实现大写字母编号的有序

列表，第 24～31 行实现小写字母和大写罗马混合编号的有序列表，由于第 26 行设置了列表项的 type 属性为"I"、value 属性为"5"，致使当前列表项前的编号变成大写罗马字母，开始顺序为"V"，大写罗马字母中第 5 个正好是 V。从第 3 个列表项开始向后所有列表项的编号顺序随之发生改变，顺序从第 6 个小写字母 f 开始向后连续编号，分别是 f、g、h、i。

## 4.4 列 表 嵌 套

在一个列表中嵌入另一个列表，作为此列表的一部分，叫列表嵌套。有序列表、无序列表可以混合嵌套，浏览器都能够自动地嵌套排列。

使用列表嵌套不仅使网页的内容布局更加合理美观，而且使其内容看起来更加简洁。列表嵌套的方式分无序列表的嵌套、有序列表的嵌套，还可以是无序列表和有序列表的混合嵌套。列表嵌套不能交叉嵌套。如<ul><ol></ul></ol>就是错误的嵌套。当然定义列表也可以与无序列表、有序列表进行嵌套。

**1. 基本语法**

```
<ul>                <!--  无序列表中嵌套有序列表  -->
    <li>项目名称
        <ol>            <!--  有序列表中又嵌套无序列表 -->
           <li>项目名称 </li>
           <li>项目名称
              <ul>
                 <li>项目名称 </li>
                 <li>项目名称 </li>
                 ...
              </ul>
           </li>
           <li>项目名称 </li>…
        </ol>
    </li>
    <li>项目名称 </li>
    <li>项目名称 </li>
</ul>
```

**【例 4-4-1】**有序列表和无序列表嵌套的应用。代码如下所示，页面效果如图 4-3 所示。

视频讲解

```
<!-- edu_4_4_1.html -->
<!doctype html>
<html lang="en">
    <head>
        <meta charset="UTF-8">
        <title>有序列表和无序列表嵌套</title>
    </head>
    <body>
        <h4>清华大学出版社图书分类</h4>
        <ol type="1">
            <li><h4>计算机与电子信息</h4>
```

```
12                <ol type="A">
13                    <li>数据库</li>
14                    <li>电子信息</li>
15                    <li>计算机组成与原理</li>
16                    <li>计算机基础
17                        <ul type="disc">
18                            <li>计算机文化基础</li>
19                            <li>公共基础</li>
20                            <li>软件技术基础</li>
21                            <li>计算机导论</li>
22                            <li>计算思维</li>
23                        </ul>
24                    </li>
25                </ol>
26            </li>
27            <li><h4>理工</h4></li>
28            <li><h4>经管与人文</h4></li>
29        </ol>
30    </body>
31 </html>
```

**2. 代码解释**

代码中第 10～29 行定义有序列表，第 12～25 行在有序列表中嵌套了 1 个有序列表，第 17～23 行又在有序列表中嵌套了 1 个无序列表。

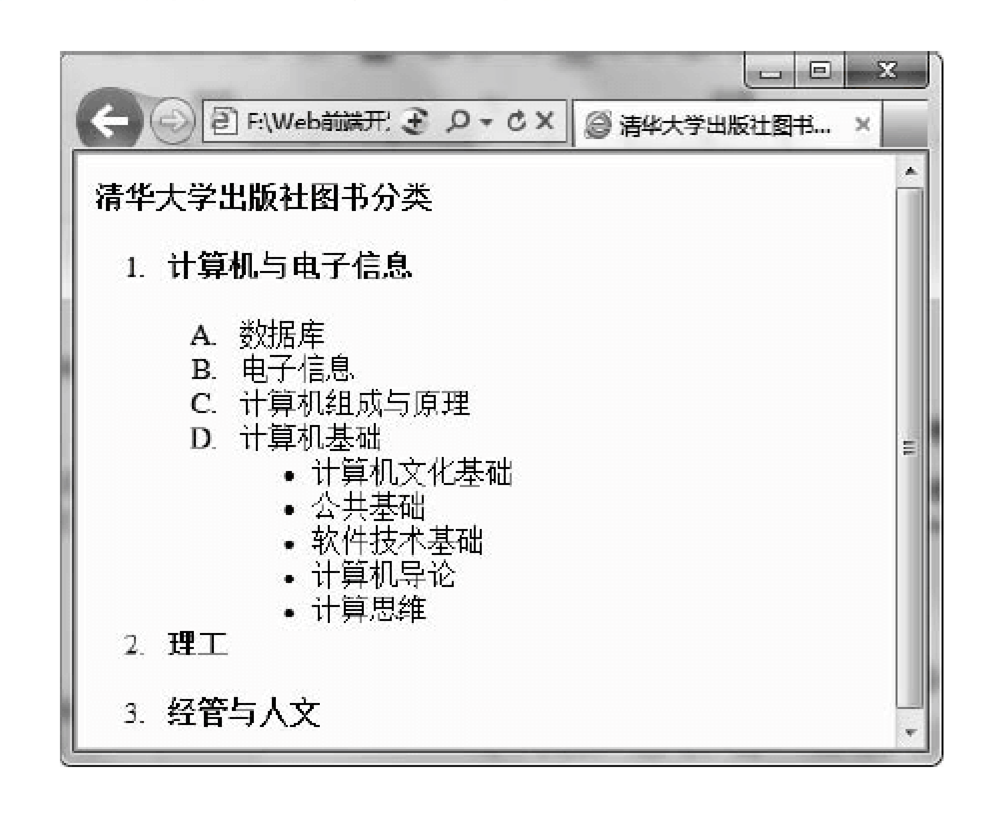



图 4-3　清华大学出版社图书分类

## 4.5　定 义 列 表

定义列表 dl（definition list）标记是成对标记，以<dl>为首标记，以</dl>为尾标记。定义列表由 dt（definition term）标记和 dd（definition description）标记组成。定义列表中每一个元素的标题使用<dt>…</dt>标记定义；后面跟随<dd>…</dd>标记，用于描述列表中元素的内容。

### 1. 基本语法

```
<dl>
    <dt>项目1</dt>
        <dd>描述1</dd>
        <dd>描述2</dd>
        <dd>描述3</dd>
    <dt>项目2</dt>
        <dd>描述1</dd>
        <dd>描述2</dd>
        ...
    <dt>项目n</dt>
</dl>
```

### 2. 语法说明

在网页中每一个 dt 标记可由一个或多个 dd 标记组成。这两个标记只能在 dl 标记中使用。定义列表的每一列表项前面既没有符号，也没有编号。

【例 4-5-1】定义列表展示联系人信息。代码如下所示，页面效果如图 4-4 所示。

视频讲解

```
<!-- edu_4_5_1.html -->
<!doctype html>
<html lang="en">
    <head>
        <meta charset="UTF-8">
        <title>定义列表</title>
    </head>
    <body>
        <h4>定义列表展示联系人信息</h4>
        <dl>
            <dt>联系人:</dt>
                <dd>张有为之</dd>
                <dd>电话：010-11011011</dd>
                <dd>E-mail: xyz@sina.com</dd>
            <dt>联系地址:</dt>
                <dd>上海市复旦大学计算机系10计算机班</dd>
            <dt>邮政编码:</dt>
                <dd>200433</dd>
        </dl>
    </body>
</html>
```




图 4-4　定义列表展示联系人信息

**3. 代码解释**

代码中第 10～19 行定义了定义列表，第 11 行、第 15 行、第 17 行定义了列表项的标题，第 12～14 行、第 16 行、第 18 行定义了列表项的描述。

## 4.6 综 合 实 例

视频讲解

以“百度糯米”美食服务项目为例，设计简易网站首页，页面效果如图 4-5 所示。

```
1  <!-- edu_4_6_1.html -->
2  <!doctype html>
3  <html lang="en">
4      <head>
5          <meta charset="UTF-8">
6          <title>多种列表在网页中使用</title>
7      </head>
8      <body>
9          <h4>百度糯米-便宜实惠，品质保证，服务专业！</h4>
10         <img src="baidunuomei.png" width="541" height="85" border="0" alt="">
11         <ul>
12             <li>麻辣烫/冒菜</li>
13             <li>美食
14                 <ul>
15                     <li>中餐/家常菜</li>
16                     <li>夏日饮品
17                         <ul>
18                             <li>米芝莲：24元</li>
19                             <li>沪上阿姨：12.90元</li>
20                             <li>哆哆鲜奶吧：30元</li>
21                             <li>黄记玉米汁：80元</li>
22                         </ul>
23                     </li>
24                 </ul>
25             </li>
26             <li>创意菜/私房菜</li>
27         </ul>
28         <dl>
29             <dt>联系客服人员:</dt>
30                 <dd>邮箱：nuomihelp@baidu.com</dd>
31                 <dd>周一至周日 9:00-22:00</dd>
32             <dt>客服电话 免长途费</dt>
33                 <dd>4006-888-887</dd>
34         </dl>
35     </body>
36 </html>
```

上述代码中第 11～27 行定义了嵌套的无序列表，将美食分为中餐/家常菜、夏日饮品，其中夏日饮品分为米芝莲、沪上阿姨、哆哆鲜奶吧、黄记玉米汁。第 28～34 行定义了定义列表，用于显示客服人员信息；第 10 行采用 img 标记插入一张百度糯米 logo 图（baidunuomei.png）。其中无序列表多层嵌套时，每层列表项前面的符号会自动变化，依次为“●”“○”“■”。

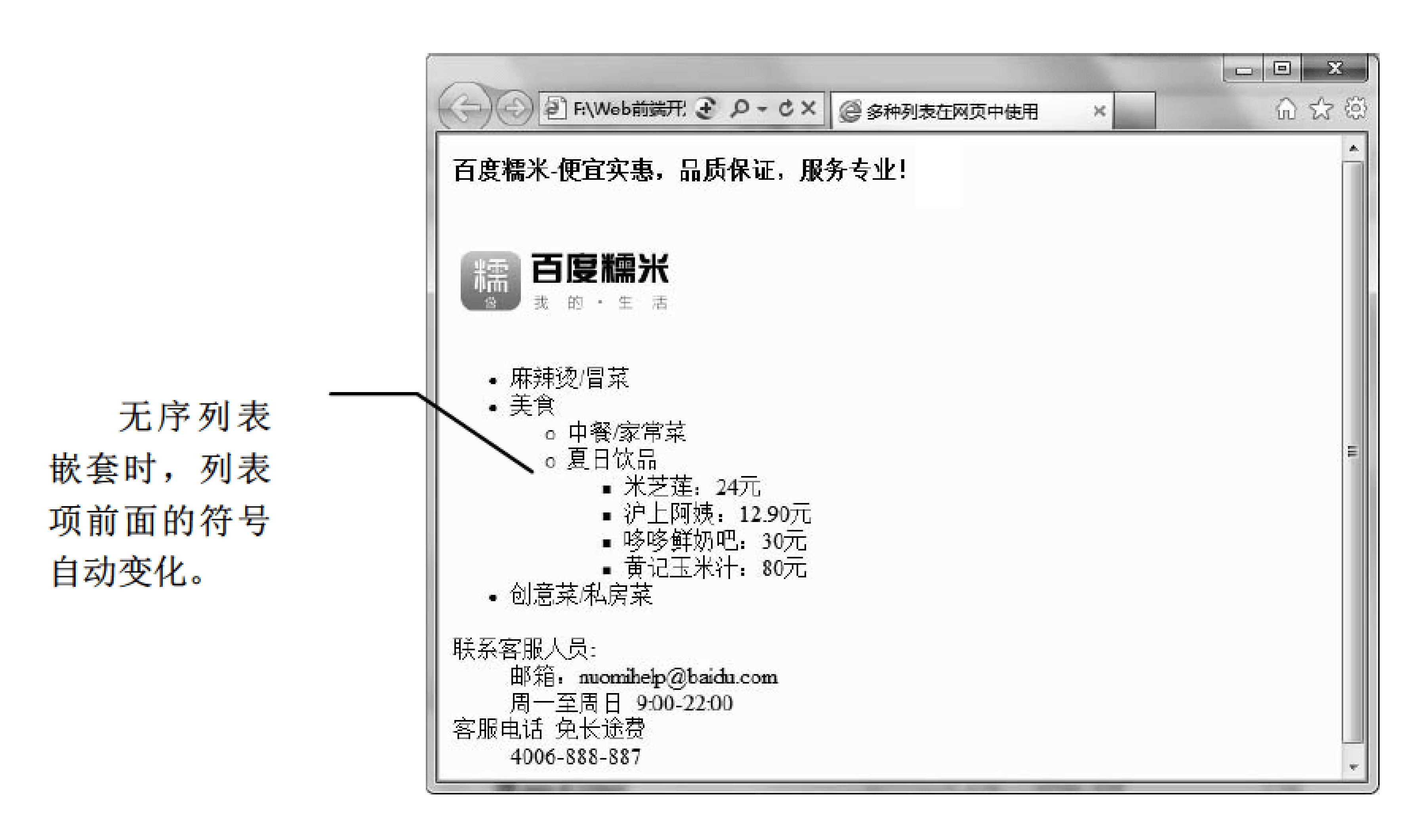



图 4-5　多种列表的应用

# 本 章 小 结

本章介绍了五种类型 HTML 列表，分别是无序列表、有序列表、定义列表、菜单列表和目录列表。但常用的列表只有三种，分别是无序列表、有序列表、定义列表。菜单列表和目录列表可以认为是无序列表的特例。

列表可以嵌套，但不能交叉嵌套，否则会发生语法错误。列表可以由无序列表和有序列表的多层子列表构成，从而使得网页内容的呈现更具层次感和美观感。

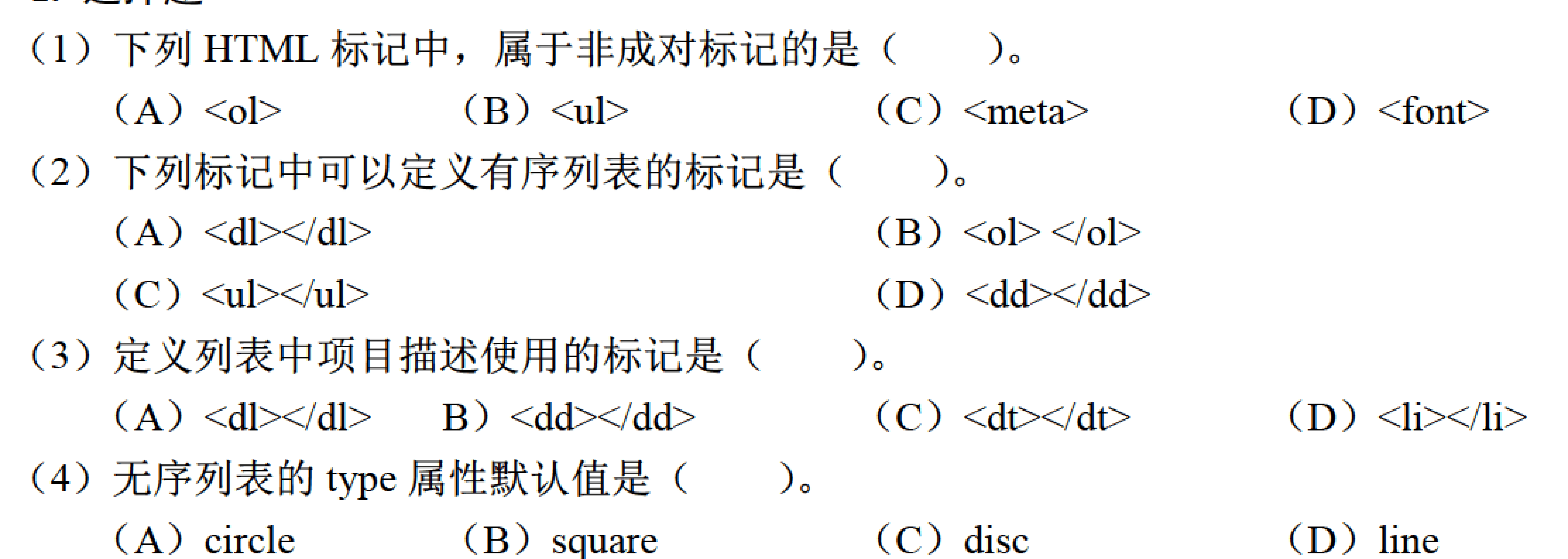
无序列表的列表项有项目符号（三种），有序列表的列表项有项目编号（五种），定义列表项目前既没有编号，也没有符号。

# 练习与实验

## 练习 4



**1. 选择题**



（1）下列 HTML 标记中，属于非成对标记的是（　　）。

（A）<ol>　　（B）<ul>　　（C）<meta>　　（D）<font>

（2）下列标记中可以定义有序列表的标记是（　　）。

（A）<dl></dl>　　（B）<ol> </ol>

（C）<ul></ul>　　（D）<dd></dd>

（3）定义列表中项目描述使用的标记是（　　）。

（A）<dl></dl>　　B）<dd></dd>　　（C）<dt></dt>　　（D）<li></li>

（4）无序列表的 type 属性默认值是（　　）。

（A）circle　　（B）square　　（C）disc　　（D）line

（5）有序列表的编号种类有（　　）个。

（A）3　　（B）4　　（C）1　　（D）5

**2. 填空题**

（1）在 HTML 文件中，ul 标记之间必须使用<li></li>标记的作用是__________。

（2）在 HTML 文件中，常用的列表有_________、_________及定义列表。

（3）设置有序列表的_______属性可以改变编号的起始值，该属性值的类型是_______，表示从哪一个数字或字母开始编号。设置列表项的_______属性后可以使该项目前面的编号发生变化，但后续的列表项前面的编号类型仍遵循原来的编号规则，只是顺序发生了改变。

**3. 简答题**

（1）简述列表类型及常用列表有哪些。

（2）简述定义列表与无序列表、有序列表的差异。

（3）简述无序列表与有序列表外在表现的差异。

## 实验 4

1. 编写代码实现 Windows 操作系统的各种版本的展示，如图 4-6 所示。

2. 编写代码实现“第四届中国大学出版社图书奖公示”页面，如图 4-7 所示。要求如下：

（1）页面标题为：“第四届中国大学出版社图书奖公示”；

（2）页面内容：2 号标题标记显示“第四届中国大学出版社图书奖公示”，页面背景色为“#ccffcc”，按图效果完成页面设计。

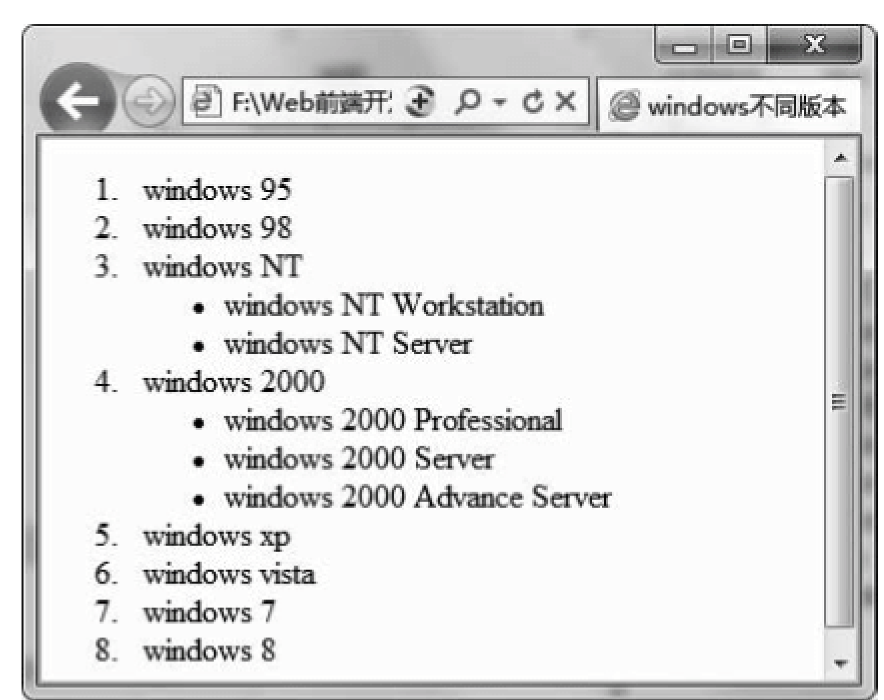



图 4-6　Windows 不同版本页面

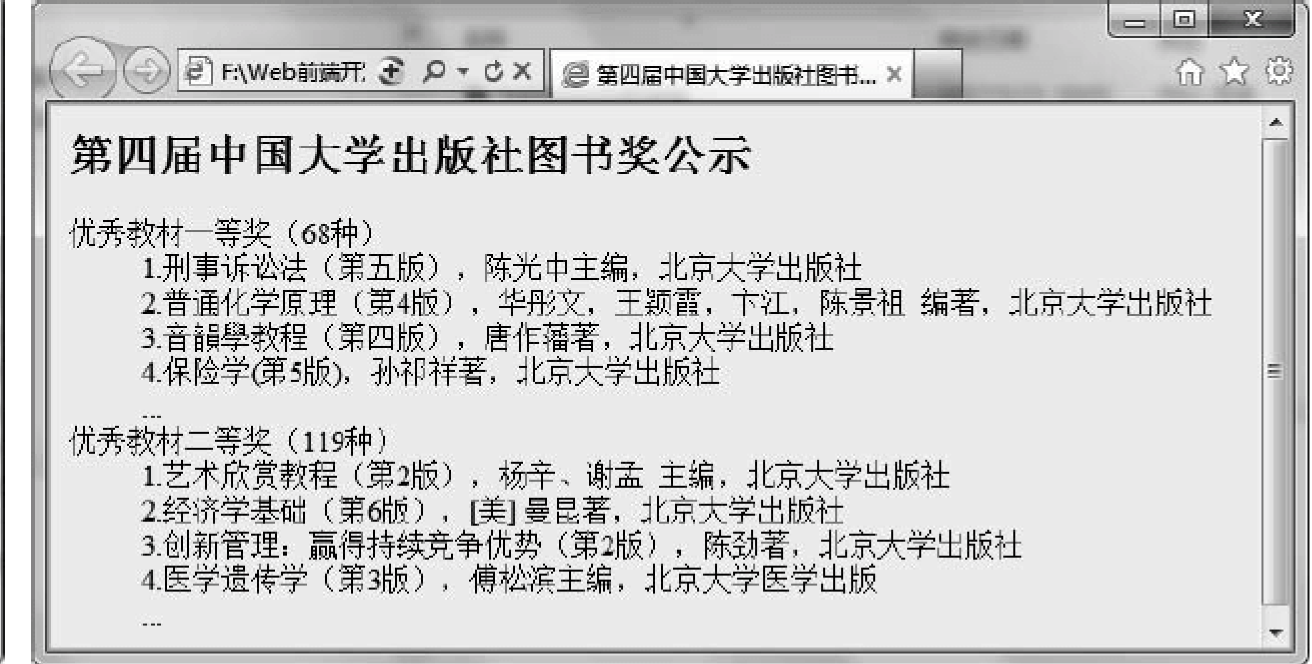



图 4-7　第四届中国大学出版社图书奖公示

# 第 5 章　超链接与浮动框架

本章学习目标

**通过超链接和浮动框架等知识的学习，能够掌握超链接和浮动框架的语法和创建方法，理解超链接的分类、路径等相关概念，学会利用超链接设置书签、文件下载、FTP 下载、电子邮件等，会使用超链接与浮动框架关联设计 Web 网页导航。**

**Web 前端开发工程师应掌握以下内容：**

- **掌握超链接的基本标记语法和属性语法。**
- **理解超链接分类、路径、书签等概念。**
- **学会使用超链接实现文件下载、FTP 下载、电子邮件链接、图像链接。**
- **学会使用超链接制作书签。**
- **学会使用浮动框架实现内嵌页面的显示。**

## 5.1　超链接概述

有了超链接，使得各个独立的网页可以有机地链接在一起构成一个网站。所谓超链接，是指从一个网页指向一个目标的连接关系，这个目标可以是另一个网页，也可以是相同网页上的不同位置，还可以是一个图片、一个电子邮件地址、一个文件，甚至是一个应用程序。用户通过浏览器浏览网页，打开页面上的超链接后，可能访问新的页面上的内容。例如百度首页，如图 5-1 所示，单击网页中“新闻”链接，会跳转到百度新闻的首页。




图 5-1　百度首页和百度新闻页面

## 5.2　超链接语法、路径及分类

### 5.2.1　超链接语法

在网页文件中，超链接通常使用链接 a 标记的超文本引用 href（Hypertext Reference）

属性建立目标对象，当前文档便是链接源，href 设置的属性值是目标文件。

### 1. 基本语法

```
<a href="url" name="" title="提示信息" target="窗口名称">超链接标题</a>
```

### 2. 语法说明

超链接 a 标记以<a>开始，以</a>结束。其间内容为超链接标题。超链接由目的地址、链接标题、打开位置三部分组成。

### 3. 属性说明

- href：链接指向的目标文件。
- name：规定锚（anchor）的名称。
- title：指向链接的提示信息。
- target：指定打开的目标窗口，如表 5-1 所示。

表 5-1 target 属性、值及说明表

| 属 性 值 | 说 明 |
|---|---|
| _self | 在当前框架中打开链接 |
| _blank | 在一个全新的空白窗口中打开链接 |
| _top | 在顶层框架中打开链接，也可以理解为在根框架中打开链接 |
| _parent | 在当前框架的上一层打开链接 |
| framename | 在指定的框架或浮动框架内打开链接，框架名称可以自定义 |

【例 5-2-1】超链接的应用。代码如下所示，页面效果如图 5-2 所示。



视频讲解

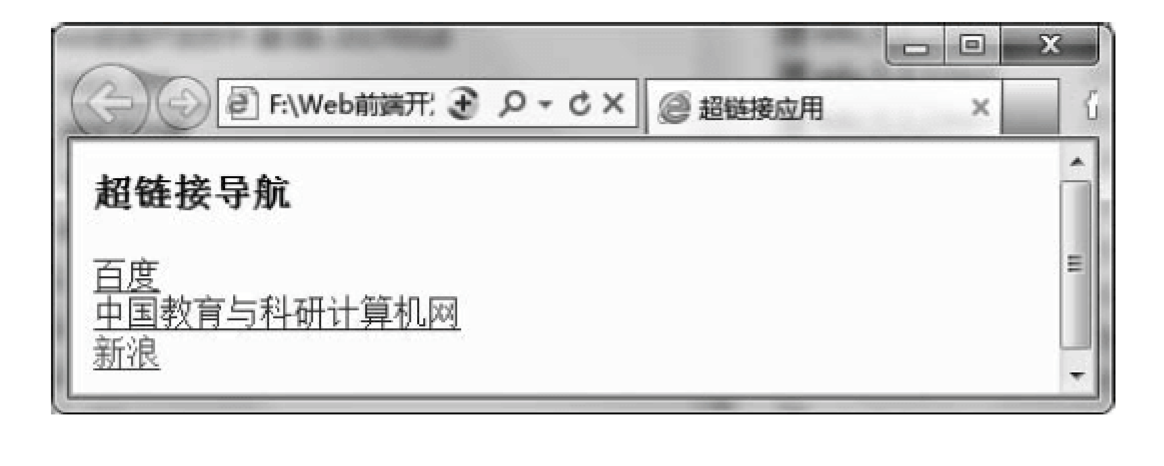



图 5-2 超链接的应用

```
<!-- edu_5_2_1.html -->
<!doctype html>
<html lang="en">
    <head>
        <meta charset="UTF-8">
        <title>超链接应用</title>
    </head>
    <body>
        <h3>超链接导航</h3>
        <a href="http://www.baidu.com" title="BaiDu">百度</a><br>
        <a href="http://www.edu.cn" target="_blank" title="CERNET">中国教育与科研计算机网</a><br>
        <a href="http://www.sina.com.cn" target="_self" title="Sina">新浪</a>
    </body>
</html>
```

**4. 代码解释**

代码中第 10～12 行在主体 body 标记插入三个超链接标记，分别设置了超链接的 href、title、target 属性。当光标移到超链接“中国教育与科研计算机网”时，会弹出提示信息“CERNET”，如图 5-2 所示。通常文本超链接的标题会显示带下画线的蓝色文本。

### 5.2.2 超链接路径

在网页设计中超链接 a 标记的 href 属性定义链接所访问的目标地址，也称为访问路径。每一个网页都有一个相对固定的地址，即统一资源定位符 URL，通过独立的 URL 可以访问不同网站上的不同的页面。在 HTML 文件中提供了三种路径，分别是绝对路径、相对路径、根路径。

**1. 绝对路径**

绝对路径指文件的完整路径，包括盘符或文件传输的协议 http、ftp 等，一般用于网站的外部链接。绝对路径有两种：一种是从盘符开始定义的文件路径，如 E:\web\index.html；一种是从协议开始定义的 URL 网址，例如中国教育与科研计算机网的网址 http://www.edu.cn。

**2. 相对路径**

相对路径是指相对于当前文件的路径，从当前文件所在位置指向目的文件的路径。采用相对路径是建立两个文件之间的相互关系，相对路径一般用于网站内部链接。相对位置的输入方法如表 5-2 所示。

**表 5-2 相对位置的输入方法**

| 相 对 位 置 | 输 入 方 法 | 代 码 示 例 |
|---|---|---|
| 同一目录 | 输入要链接的文档 | <a href="tongzhi.html">通知</a> |
| 链接上一目录 | 先输入“../”，再输入目录名 | <a href="../index.html">首页</a> |
| 链接下一目录 | 先输入目录名，后加“/” | <a href="ks/note.html">考试通知</a> |

**3. 根路径**

根路径是指从网站的最底层开始起，一般网站的根目录就是域名下对应的文件夹，例如 E 盘上存放一个网站，双击 E 盘进入到 E 盘看到的就是网站的根目录，这种路径就称为根路径，所以根路径以斜杠/开头，然后书写文件夹名，接着书写子文件夹名或文件名，以此类推，直到写完路径为止。如：/web/news/show.html。根目录需要带盘符时，采用格式为 E：/web/news/show.html，这表示 E 盘下 web/news 下的 show.html。这种写法在 Google 的 Chrome、Firefox 等浏览器能够支持。不过 IE 浏览器、360 安全浏览器不支持这种写法，正确写法如下所示：

```
<a href="d:/web/news/show.html">显示信息</a>
```

根路径一般用于创建内部链接。通常不建议采用此种链接形式。

根目录路径和相对路径都是以某个位置为起点的相对路径，但是根目录路径一般用于有多台服务器的大型网站，建议对路径的概念不大熟悉的初学者在做链接时还是采用相对路径为宜。另外，为了避免链接错误的出现，不管使用何种类型的链接，站点的建立是必要的。

### 5.2.3　超链接分类

在 HTML 文件中，超链接可以分为内部链接和外部链接两种。内部链接是指网站内部文件之间的链接，而外部链接是指网站内的文件链接到站点内容外的文件。将 URL 设置为相对路径为内部链接，而将 URL 设置为文件的绝对路径则为外部链接。

【例 5-2-2】内部链接和外部链接的应用。代码如下所示，页面效果如图 5-3 所示。

视频讲解

```
1 <!-- edu_5_2_2.html -->
2 <!doctype html>
3 <html lang="en">
4     <head>
5         <meta charset="UTF-8">
6         <title>内部链接和外部链接</title>
7     </head>
8     <body>
9         <h2>内部链接：</h2>
10        <p><a href="index.html">通知</a>指向网站内的页面链接</p>
11        <h2>外部链接：</h2>
12        <p><a href="http://www.163.com/">网易</a>指向网站外的页面链接</p>
13    </body>
14 </html>
```

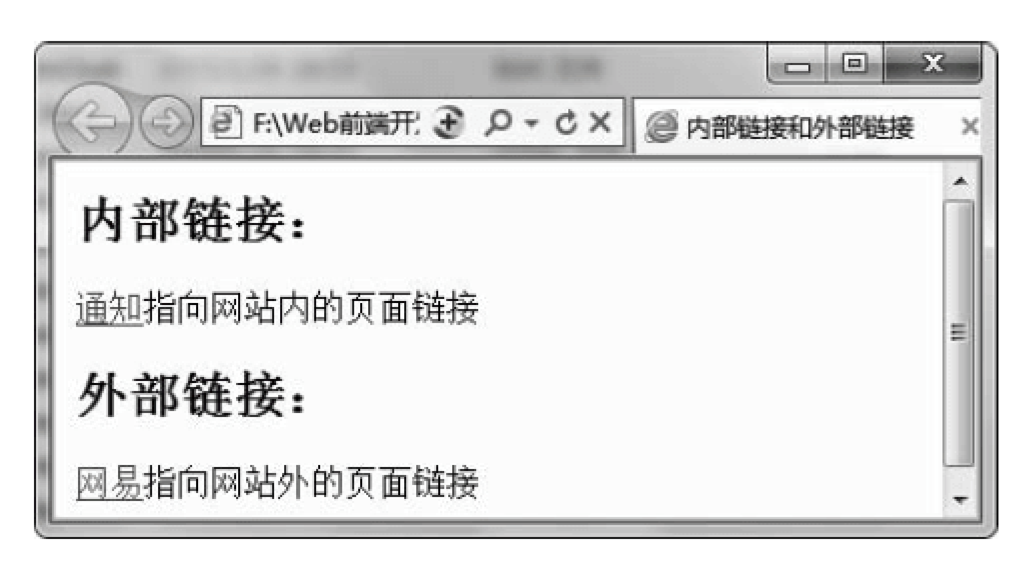



图 5-3　内部链接和外部链接的应用

代码中第 10 行定义访问当前目录下 index.html 的内部链接，第 12 行定义访问网易网站的外部链接。

## 5.3　超链接的应用

在网络上能够通过链接，访问不同的资源或网页。链接对象也是多种多样，可分为文件、FTP 站点、图像、电子邮件及书签等。

### 5.3.1　创建 HTTP 文件下载超链接

网站经常提供软件、文件等资料下载，下载文件的链接指向文件所在的相对路径或绝对路径，文件类型为*.doc、*.pdf、*.exe、*.rar 等。基本语法如下：

```
<a href="url">链接内容</a>
```

### 5.3.2　创建 FTP 站点访问超链接

FTP 服务器链接和网页链接区别在于所用协议不同，浏览网页采用 http 协议，而访问

FTP 服务器采用 FTP 协议连接。FTP 需要从服务器管理员处获得登录的权限。不过部分 FTP 服务器可以匿名访问，从而能获得一些公开的文件。基本语法如下：

```
<a href = "ftp://服务器IP地址或域名">链接的文字</a>
```

### 5.3.3　创建图像超链接

将链接标题替换为一幅图像，浏览时单击该图像，就可以打开链接目标文件。基本语法如下：

```
<a href="URL"><img src="" alt=""></a>
```

使用<img>标记替代原来超链接的标题，即可实现图像链接。

### 5.3.4　创建电子邮件超链接

一般网站上都会设置“联系我们”这样的栏目或超链接，目的是方便用户及时与网站管理员进行沟通与联系，这就是常说的电子邮件链接。基本语法如下：

```
<a href="mailto:E-mail地址[ ?subject=邮件主题[&参数=参数值]] ">链接内容</a>
```

邮件地址必须完整，例如 intel@qq.com。参数有 cc（抄送）、bcc（密送）、subject（主题）、body。多个收件人用分号“;”分隔；多个参数用“&”连接，“&”与关键字之间不能留有空格；空格用“%20”替代。

**举例如下：**

```
<a href="mailto:some@mysoft.com;jlchu@163.com?cc=xyz@163.com
&bcc=anbo@sina.com&subject=Hello%20again&body=下周二开会讨论">发送邮件</a>
```

**【例 5-3-1】**超链接的综合应用。代码如下所示，页面效果如图 5-4～图 5-8 所示。

视频讲解

```
<!-- edu_5_3_1.html -->
<!doctype html>
<html lang="en">
   <head>
      <meta charset="UTF-8">
      <title>超链接的应用</title>
      <style type="text/css">
         h1{background:#9999cc;color:white;padding:10px;height:50px;
         text-align:center;}
         img{width:70px;height:45px;}
      </style>
   </head>
   <body>
      <h1>超链接的应用</h1>
      <h3><a href="ch5.ppt">1.HTTP下载-文件ch5.ppt</a></h3>
      <h3><a href="ftp:// ftp.pku.edu.cn">2.FTP下载-北京大学FTP站点</a>
      </h3>
      <h3>3.图像超链接
      <a href="https://www.baidu.com//">
         <img border="0" src="bd_logo1.png" />
      </a></h3>
      <h3>4.邮件超链接-有问题可以给我
```

```
21          <a href="mailto:someone@microsoft.com;xyz@163.com?cc=jlchu@163.
            com&bcc=12345678@qq.com&subject=Hello%20again&body=下午14:20开
            会">发送邮件</a></h3>
22          <p>应该使用%20来替换单词之间的空格，这样浏览器就可以正确地显示文本了。</p>
23      </body>
24  </html>
```

**代码解释**

代码中第 14 行定义 ch5.ppt 文件的 HTTP 下载链接，单击超链接进入文件下载页面，如图 5-5 所示。第 15 行定义访问北京大学 FTP 站点的 FTP 下载链接，单击超链接进入 FTP 站点下载文件页面，如图 5-6 所示。第 16～19 行定义图像超链接，单击图像超链接进入百度搜索页面，如图 5-7 所示。第 20～21 行定义邮件超链接，单击邮件超链接进入邮件设置页面，如图 5-8 所示。




图 5-4　超链接的应用




图 5-5　HTTP 下载链接

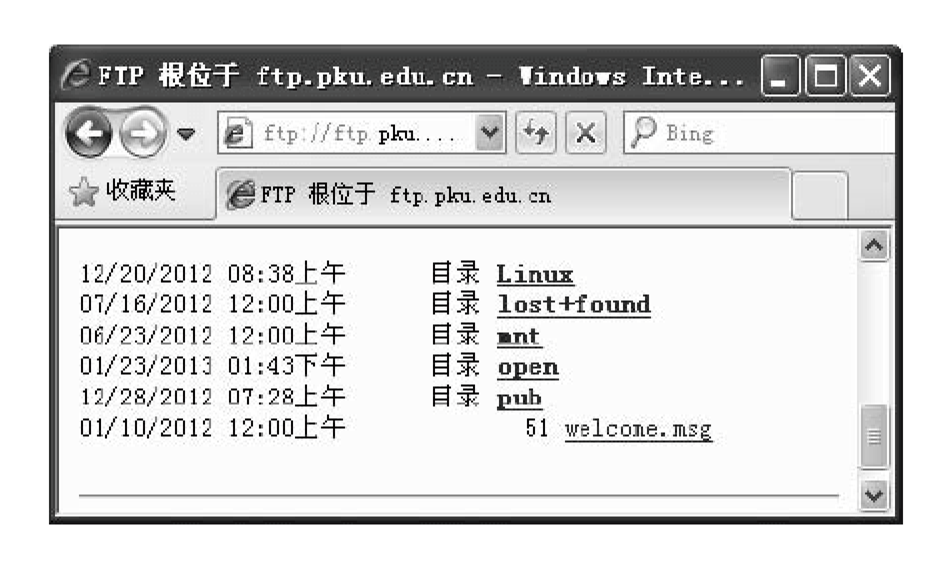



图 5-6　FTP 下载链接




图 5-7　图像超链接




图 5-8　邮件超链接

## 5.3.5 创建页面书签链接

书签是指到文章内部的链接，可以实现段落间的任意跳转。实现这样的链接要先定义一个书签作为目标端点，再定义到书签的链接。链接到书签分为两种：链接到同一页面中的书签和链接到不同页面中的书签。

**1. 定义书签**

通过设置超链接 a 标记的 name 属性来定义书签。

```
<a name="书签名">书签标题</a>
```

name 属性的值是定义书签的名称，供书签链接引用的。超链接<a></a>之间的信息为书签的标题。

**2. 定义书签链接**

通过设置超链接 a 标记的 href 属性来定义书签链接。

1）基本语法

```
<a href="#书签名">书签标题</a>        <!-- 同一页面内   -->
<a href="URL#书签名">书签标题</a>   <!--  不同页面间  -->
```

第一种是对于同一页面内的书签，第二种是不同页面间的书签，其中 URL 设置 HTML 文件名称，“#书签名”表示引用名称为“书签名”的书签。

【例 5-3-2】书签链接的应用。编写 edu_5_3_2.html 和 edu_5_3_2_1.html，代码如下所示，页面效果如图 5-9 和图 5-10 所示。

视频讲解

```
<!-- edu_5_3_2.html -->
<!doctype html>
<html lang="en">
   <head>
       <meta charset="UTF-8">
       <title>链接到同一页面的书签</title>
   </head>
   <body>
       <h3><a name="software">主流的网页设计软件</a></h3>
       <ul>
           <li><a href="#dw">Dreamweaver MX[同页]</a></li>
           <li><a href="#fl">Flash MX[同页]</a></li>
           <li><a href="#fw">Fireworks MX[同页]</a></li>
           <li><a href="edu_5_3_2_1.html#EditPlus">EditPlus[异页]</a>
          </li>
       </ul>
       <h2><a name="dw">Dreamweaver MX</a></h2>
       <p>    Dreamweaver是美国Macromedia公司(现已被
      Adobe公司收购，成为Adobe Dreamweaver）开发的集网页制作和管理网站于一身
      的所见即所得网页编辑器，它是第一套针对专业网页设计师特别发展的视觉化网页开发
      工具，利用它可以轻而易举地制作出跨越平台限制和跨越浏览器限制的充满动感的网
      页。</p>
       <h4 align="right"><a href="#software">返回</a></h4>
```

```
        <h2><a name="fl">Flash MX</a></h2>
        <p>    Flash是美国Macromedia公司所设计的二维动画软件，全称Macromedia Flash（被Adobe公司收购后称为Adobe Flash），主要用于设计和编辑Flash文档。附带的Macromedia Flash Player，用于播放Flash文档。
        现在，Flash已经被Adobe公司购买，最新版本为：Adobe Flash CS6 Professional，播放器也更名为Adobe Flash Player。</p>
        <h4 align="right"><a href="#software">返回</a></h4>
        <h2><a name="fw">Fireworks MX</a></h2>
        <p>    Adobe Fireworks可以加速 Web 设计与开发，是一款创建与优化 Web 图像和快速构建网站与 Web 界面原型的理想工具。Fireworks不仅具备编辑矢量图形与位图图像的灵活性，还提供了一个预先构建资源的公用库，并可与 Adobe Photoshop、Adobe Illustrator、Adobe Dreamweaver和Adobe Flash软件进行集成。在Fireworks中将设计迅速转变为模型，或利用来自Illustrator、Photoshop 和 Flash 的其他资源。然后直接置入Dreamweaver 中轻松地进行开发与部署。</p>
        <h4 align="right"><a href="#software">返回</a></h4>
    </body>
</html>
```

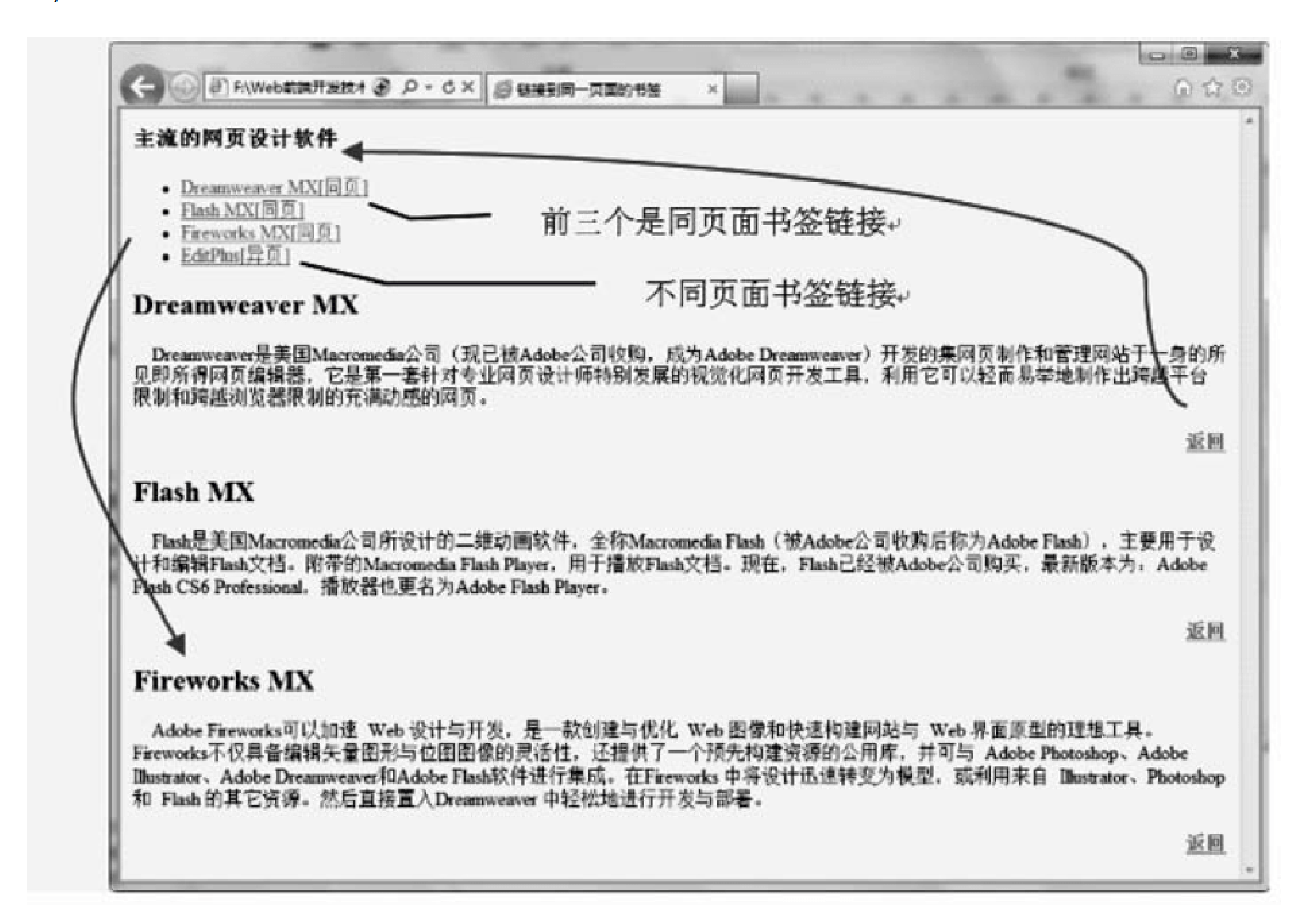



图 5-9　同页面书签链接

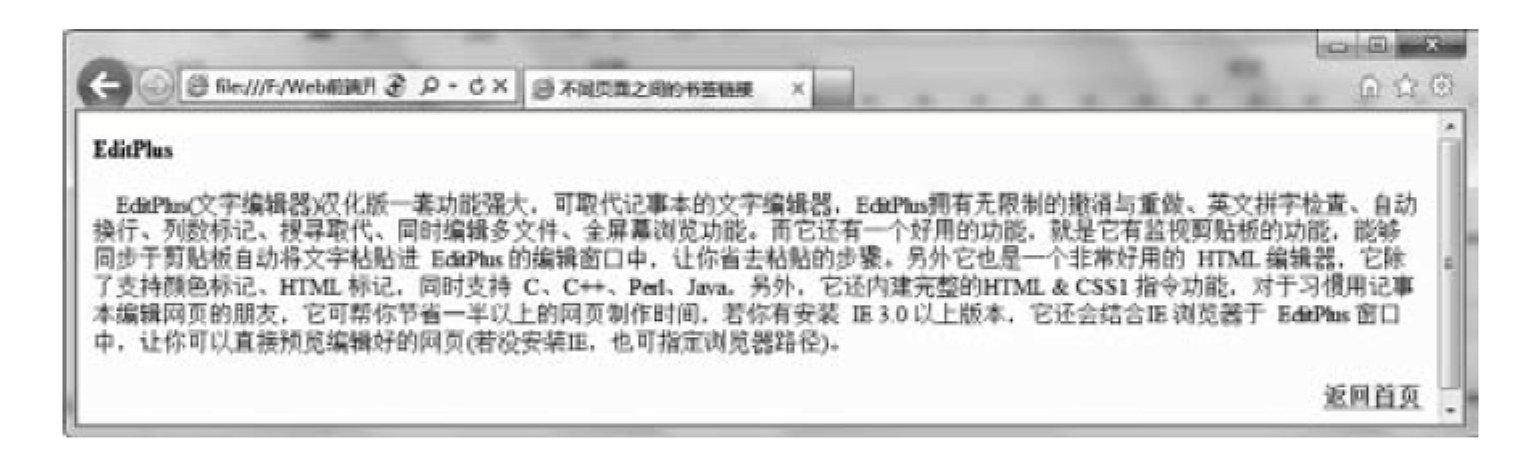



图 5-10　不同页面间书签链接

2）代码解释

代码中第 9 行定义根书签名称为"software"，供所有的“返回”书签链接引用；第 10～15 行利用无序列表定义四个书签链接，其中前三个为同页面书签链接，最后一个为不同页面的书签链接，跳转到 edu_5_3_2_1.html 页面上访问书签 EditPlus；第 16 行、第 19 行、第 23 行定义三个书签分别为"dw"、"fl"、"fw"，供第 11 行、第 12 行、第 13 行的定义的书签链接引用；第 18 行、第 22 行、第 25 行定义“返回”书签链接，返回根书签所在位置。单击图 5-9 中的“EditPlus[异页]”访问 edu_5_3_2_1.html 页面上的 EditPlus 书签，如图 5-10 所示。

```
 1 <!-- edu_5_3_2_1.html -->
 2 <!doctype html>
 3 <html lang="en">
 4     <head>
 5         <meta charset="UTF-8">
 6         <title>不同页面之间的书签链接 </title>
 7     </head>
 8     <body>
 9         <h4><a name="EditPlus">EditPlus</a></h4>
10         <p>    EditPlus（文字编辑器）汉化版一套功能强大，
           可取代记事本的文字编辑器，EditPlus拥有无限制的撤销与重做、英文拼字检查、自
           动换行、列数标记、搜寻取代、同时编辑多文件、全屏幕浏览功能。而它还有一个好用
           的功能，就是它有监视剪贴板的功能，能够同步于剪贴板自动将文字粘贴进 EditPlus
           的编辑窗口中，让你省去粘贴的步骤。另外它也是一个非常好用的 HTML 编辑器，它
           除了支持颜色标记、HTML 标记，同时支持 C、C++、Perl、Java，另外，它还内建
           完整的HTML & CSS1 指令功能，对于习惯用记事本编辑网页的朋友，它可帮你节省一
           半以上的网页制作时间，若你有安装IE 3.0 以上版本，它还会结合IE 浏览器于
           EditPlus 窗口中，让你可以直接预览编辑好的网页（若没安装IE，也可指定浏览器路径）。
11         </p>
12         <h4 align="right"><a href=" edu_5_3_2.html#software">返回首页
           </a></h4>
13     </body>
14 </html>
```

上述代码中第 9 行在标题字 h4 标记内定义书签名称为"EditPlus"，作为 edu_5_3_2.html 页面的书签链接的目标。第 12 行定义返回 edu_5_3_2.html 页面的书签链接，单击“返回首页”返回 edu_5_3_2.html 页面，如图 5-9 所示。

## 5.4　浮 动 框 架

浏览器窗口含有孤立的子窗口称为浮动框架，也称为内联框架。在浏览器窗口中嵌入浮动框架可使用 iframe 标记，该标记为成对标记，必须插入在 body 标记中，而不能插入到 frameset 标记中。

### 1. 基本语法

```
<iframe 属性名称="value" name="iframename"></ iframe>
<a href="target.html" target="iframename " >链接标题</a>
```

**2．语法说明**

属性名称及相关说明如表 5-3 所示。

表 5-3　浮动框架属性

| 属　性 | 说　明 | 属　性 | 说　明 |
|---|---|---|---|
| src | 设置源文件属性 | frameborder | 设置框架边框 |
| name | 设置框架名称 | scrolling | 设置框架滚动条 |
| width | 设置浮动框架窗口宽度 | marginwidth | 设置框架左右边距 |
| height | 设置浮动框架窗口高度 | marginheight | 设置框架上下边距 |

【例 5-4-1】应用浮动框架。代码如下所示，页面效果如图 5-11 和图 5-12 所示。

视频讲解

```
1 <!-- edu_5_4_1.html -->
2 <!doctype html>
3 <html lang="en">
4     <head>
5         <meta charset="UTF-8">
6         <title>浮动框架应用</title>
7         <style type="text/css">
8             a{width:300px;margin:0 10px;}
9             h3{font-size:28px;color:#0000ff;text-align:center;}
10            div{margin:0 auto;text-align:center;}
11        </style>
12    </head>
13    <body>
14        <div id="" class="">
15            <h3>浮动框架应用</h3>
16            <hr color="red">
17            <iframe name="leftiframe" src="http://www.njust.edu.cn"
              width="300" height="300" ></iframe>
18              
19            <iframe name="rightiframe" src="http://www.pku.edu.cn"
              width="300" height="300" marginwidth="10px"></iframe>
20            <p>
21                <a href="http://www.gov.cn" target="leftiframe">在左边浮动
                  框架内显示中央人民政府网站</a>
22                <a href="http://www.moe.gov.cn/" target="rightiframe">在
                  右边浮动框架内显示教育部网站</a>
23            </p>
24        </div>
25    </body>
26 </html>
```

**3．代码解释**

代码中第 17 行在 div 标记中插入一个名称为 leftiframe 的浮动框架，并为该框架设置了内部显示的网页、宽度、高度；第 19 行在 div 标记中插入一个名称为 rightiframe 的浮动框架，并为该框架设置了内部显示的网页、宽度、高度、框架的左右边距等属性；第 21 行、第 22 行将浮动框架 leftiframe、rigtiframe 设置为超链接的链接目标。单击超链接在左、右浮动框架中分别显示不同的页面，如图 5-12 所示。




图 5-11　在浮动框架内显示指定网页的初始图




图 5-12　单击超链接后页面效果图

# 5.5 综 合 实 例



视频讲解

以“百度”首页为模板，设计百度仿真页面，效果如图 5-13 所示。




图 5-13　百度仿真页面

```
<!-- edu_5_5_1.html -->
<!doctype html>
<html lang="en">
    <head>
        <meta charset="UTF-8">
        <title>百度仿真页面</title>
    </head>
    <body>
        <p align="center"><a href="http://www.baidu.com">
        <img border="0" src="baidu_sylogo1.gif" /></a></p>
        <p align="center">
            <a href="http://news.baidu.com" name="tj_news">新 闻</a>
            <b>网 页</b>
            <a href="http://tieba.baidu.com" name="tj_tieba">贴 吧</a>
            <a href="http://zhidao.baidu.com" name="tj_zhidao">知 
            道</a>
            <a href="http://music.baidu.com" name="tj_mp3">音 乐</a>
            <a href="http://image.baidu.com" name="tj_img">图 片</a>
            <a href="http://video.baidu.com" name="tj_video">视 频</a>
            <a href="http://map.baidu.com" name="tj_map">地 图</a>
        </p>
```

```
21          <p align="center">
22             <input type="text" size="60" name="">
23             <input type="button" name="baidu" value="百度一下">
24          </p>
25          <p align="center">问题反馈请<a href="mailto:someone@baidu.com?
            subject=问题反馈">发送邮件</a></p>
26       </body>
27 </html>
```

上述代码中第 9～10 行实现百度图片链接；第 11～20 行在段落 p 标记插入七个文字超链接；第 21～24 行在段落 p 标记内分别插入文本输入框和普通按钮，用于仿真百度首页的搜索功能，第 22 行定义文本输入框长度为 60 个字符，第 23 行定义一个普通按钮；第 25 行实现电子邮件的链接。

## 本 章 小 结

本章主要学习了超链接和浮动框架的知识。重点介绍了超链接语法、超链接中路径以及与浮动框架的关联。区别使用绝对路径、相对路径及根路径设置超链接目标。理解超链接的类型及每种类型适用场合，其中内部链接用于网站内部资源之间的链接，而外部链接用于网站外部的链接。

同时本章还介绍了超链接的不同链接对象的语法和使用方法，包括下载文件链接、书签链接、FTP 链接、图像链接、电子邮件链接。

## 练习与实验

### 练习 5

**1. 选择题**

（1）下列电子邮件链接格式正确的是（　　）。

（A）<a href="mailto:xxx.com.cn?subject=你好！">...</a>
（B）<a href="mailto:xxx@.net?subject=你好！">...</a>
（C）<a href="mailto:xxx@com?subject=你好！">...</a>
（D）<a href="mailto:xxx@xxx.com?subject=你好！">...</a>

（2）当链接指向（　　）文件时，不打开该文件，而是提供给浏览器下载。

（A）ASP　　（B）HTML　　（C）ZIP　　（D）CGI

（3）下列选项中不是超链接的 target 属性取值的是（　　）。

（A）_self　　（B）_new　　（C）_blank　　（D）_top

（4）在网页中，能够定义超链接的标记是（　　）。

（A）<link>…</link>　　（B）<h1>…</h1>
（C）<a>…</a>　　（D）<ul>…</ul>

（5）<img>标记中规定图像 URL 的属性是（　　）。

（A）href　　（B）src　　（C）type　　（D）align

（6）在 HTML 中，要定义一个书签链接应该使用的语句是（　　）。

（A）<a href="#book1">text</a>　　（B）<a name="book1">text</a>

（C）<a target="#book1">text</a>　　（D）<a link="#book1">text</a>

（7）在 body 中插入浮动框架正确语句是（　　）。

（A）<body><iframe src="" name="rightframe">…</body>

（B）<body><iframe src="" name="rightframe"></iframe>…</body>

（C）<body><frame src="" name="rightframe">…</body>

（D）<body><frame src="" name="rightframe"></frame>…</body>

**2. 填空题**

（1）如果要创建一个指向电子邮件 someone@mail.com 的超链接，代码应该如下：

<a ______________________________>指向 someone@mail.com 的超链接</a>

（2）在指定页内超链接的时候，如果在某一个位置使用了<a ______ ="target1">书签</a>语句定义了书签名为 target1，那么当单击超链接<a href=__________________>书签链接</a>时，能够跳转到同页面定义的书签 target1 位置上。

（3）超链接路径分为__________、_________、__________。网站内部链接一般使用_____路径，当然______路径也可以用于内部链接；外部链接一般使用______路径。

（4）浮动框架的 name 属性值为"leftframe"，让超链接在此浮动框架中打开“中国教育网（URL：http://www.edu.cn）”网站的正确的超链接是________________________。

**3. 简答题**

（1）简述什么是绝对路径和相对路径。

（2）写出制作页面书签的步骤，并举例说明。

（3）如果想通过单击不同的超链接在浮动框架中打开不同的页面，需要如何设置？

# 实验 5

1．根据提供的图像和超链接资源完成图像页面导航设计，资源与对应的超链接如表 5-4 所示，效果如图 5-14 所示。编写符合以下要求的文档：在 HTML 文档中插入一张图片，为图片加上链接，指向它所在的网站。

表 5-4　图像与超链接对应关系表

| 序　　号 | 图 片 名 称 | URL |
|---|---|---|
| 1 | ipadblank1.gif | http://www.apple.com.cn/iphone/ |
| 2 | ipadblank2.gif | http://www.apple.com.cn/iphone/ |
| 3 | ipadblank3.gif | http://www.apple.com.cn/macbook-pro/ |
| 4 | ipadblank4.gif | http://www.apple.com.cn/supplierresponsibility/ |




图 5-14　图片超链接的页面

2．按如下要求设计 Web 页面，如图 5-15 所示。

（1）页面标题为：桂林山水风景图片。

（2）正文标题为红色“桂林山水风景图片”，图片分别为 image51.jpg、image52.jpg、image53.jpg、image54.jpg；采用无序列表布局，每一个列表项的内容为图像链接，单击小图，可以浏览大图。

（3）定义样式。img 标记样式为“宽度 100px、高度 100px、边框 0px”；h3 标记样式为“红色、居中”；ul 样式为“去除列表项前的符号、内容居中显示”；li 样式为“显示方式行内显示（display:inline）、宽度 120px、行高 30px”。




图 5-15　桂林山水风景

3．设计如图 5-16 所示页面，要求如下：

（1）页面标题为：“浮动框架应用”。

（2）页面内容为：一个标题，网页内嵌入两个浮动框架、两个超链接。标题内容：三号标题显示“浮动框架中打开新页面”；两个浮动框架名称分别为 left、right，初始网页分别为：http://www.pku.edu.cn 和 http://www.seu.edu.cn;两个超链接分别在左、右两个浮动框架中打开两个网页，分别是 https://www.baidu.com、http://www.qq.com，两个超链接的 target 属性分别指向两个浮动框架 left、right。






图 5-16　浮动框架中打开新网页

# 第 6 章　图像与多媒体文件

本章学习目标

**优秀的商用的网站往往通过为页面添加大量的图像、声音、视频、动画等多媒体信息来丰富网站的内容，吸引更多网络访问者的关注。目前大型商业网站非常注重 Web 前端开发技术的研究，通过组合各类前端开发技术来改善用户体验和增加用户互动环节，最大限度地获取商业利润。本章重点介绍图像、滚动文字、音频等多媒体文件在 HTML 文件的使用方法。**

**Web 前端开发工程师应掌握以下内容：**

- **掌握图像 img 标记语法、属性设置及图像热区链接的设置方法。**
- **掌握滚动文字 marquee 标记语法及属性设置方法。**
- **掌握嵌入多媒体文件 embed 标记语法及属性设置方法。**
- **学会采用超链接插入动画、音频和视频类等多媒体文件。**

## 6.1　图　　像

图像和多媒体文件是网页中必不可少的元素，灵活地应用这些元素会给网页增添不少色彩。而且图像及其多媒体文件的直观、明了、绚丽和美观等都是文字无法替代的。

在网页上常见的图像格式有 JPG（Joint Photo graphic Experts Group）、GIF（Graphics Interchange Format）和 PNG（Portable Network Graphic Format）等。BMP 格式不常用。

HTML 文件中使用 img 标记在网页上插入图像。设置它的属性可以控制图像的路径、尺寸和替换文字等各种功能。

### 6.1.1　插入图像

可以使用 HTML 的 img 标记将图像插入到网页中，也可以使用 CSS 设置成某元素的背景图像，而根据图像的格式不同，其适用的地方也不同。

**1. 基本语法**

```
<img src="URL" alt="替代文本">
```

**2. 语法说明**

img 标记是单（个）标记，图像样式由 img 标记的属性决定。img 标记有两个必选属性，分别是 src、alt，其他属性为可选属性，具体属性、取值及说明如表 6-1 所示。

src 指“source”。源属性的值是图像的 URL 地址。可以采用绝对路径或相对路径来

表示文件的位置，如 src="d:/web/ch6/images1.jpg"是采用绝对路径，而 src="images1.jpg"是采用相对路径。

**表 6-1　img 标记属性名、值及说明表**

| 属　　性 | 值 | 说　　明 |
|---|---|---|
| alt | text | 规定图像的替代文本 |
| src | URL | 规定显示图像的 URL |
| name | text | 规定图像的名称 |
| height | pixels、% | 定义图像的高度 |
| width | pixels、% | 设置图像的宽度 |
| align | top\|middle\| bottom \|left\|center\|right | 规定如何根据周围的文本来排列图像，分水平、垂直两个方向 |
| border | pixels | 定义图像周围的边框 |
| hspace | pixels | 定义图像左侧和右侧的空白 |
| vspace | pixels | 定义图像顶部和底部的空白 |
| usemap | URL | 将图像定义为客户器端图像映射 |

【例 6-1-1】在网页中插入图像。代码如下所示，页面效果如图 6-1 所示。

```
1 <!-- edu_6_1_1.html -->
2 <!doctype html>
3 <html lang="en">
4   <head>
5         <meta charset="UTF-8">
6         <title> 插入图像 </title>
7         <style type="text/css">
8             body{text-align:center;}
9         </style>
10   </head>
11   <body>
12       <h2>网页中插入图像</h2>
13       <hr color="#66ff33" width="60%">
14       <img src="images1.jpg" alt="机房">
15   </body>
16 </html>
```



视频讲解

### 3. 代码解释

代码中第 12～14 行定义标题字、水平分隔线、图像。其中第 14 行采用相对路径在网页中插入图像 images1.jpg，图像格式为 JPEG。第 8 行实现内容全部居中显示。




图 6-1　插入图像

## 6.1.2　设置图像的替代文本

img 标记的 alt 属性用来为图像设置替代文本。替代文本有两个作用：

- 浏览网页时，鼠标悬停在图像上，鼠标旁边会出现替代文本；
- 图像加载失效时，在图像的位置上会显示红色的“×”，并显示替代文本。

**1. 基本语法**

```
<img src=" URL " alt="替代文本">
```

**2. 语法说明**

alt 属性：替代文本既可以是中文也可以是英文。

【例 6-1-2】设置图像的替代文本。代码如下所示，页面效果如图 6-2 所示。

```
1 <!-- edu_6_1_2.html -->
2 <!doctype html>
3 <html lang="en">
4     <head>
5         <meta charset="UTF-8">
6         <title> 插入图像 </title>
7         <style type="text/css">
8         body{text-align:center;}
9         </style>
10    </head>
11    <body>
12        <h3>网页中插入图像</h3>
13        <hr color="#3300ff">
14        <img src="images1.jpg" alt="网络机房" title="网络机房">
15    </body>
16 </html>
```

视频讲解




图 6-2　添加图像替代文字

**3. 代码解释**

代码中第 14 行插入一幅图像，并设置 alt 属性值为“网络机房”，当图像成功加载时，会显示图 6-2 左图效果；当图像加载不成功时，会显示右图的效果。

## 6.1.3　设置图像的高度和宽度

img 标记的 width 和 height 属性用来设置图像的宽度和高度。默认情况下，网页中的图像大小就是由图像原来的宽度和高度来决定。如果不设置图像的宽度和高度，图像的大小和原图是一样的。

**1. 基本语法**

```
<img src=" URL " width="value" height="value">
```

**2. 语法说明**

- 图像高度和宽度的单位可以是像素，也可以是百分比。
- 在设置图像的宽度和高度的属性时，可以只设置宽度和高度中的其中之一，另一个属性将按原图像宽高等比例显示；同时设置两个属性时图像会发生变形。

### 6.1.4　设置图像的边框

默认的图像是没有边框的，通过 img 标记的 border 属性可以为图像设置边框的宽度。但边框的颜色是不可以调整的，当未设置图像链接时，边框的颜色为黑色；当设置图像链接时，边框的颜色和链接文字颜色一致，默认为深蓝色。通过样式表可修改边框的线型、宽度和颜色。

**1. 基本语法**

```
<img src=" URL " border="value">
```

**2. 语法说明**

value 为边框线的宽度，用数字表示，单位为像素。

【**例 6-1-3**】设置图像的高度、宽度及边框。代码如下所示，页面效果如图 6-3 所示。

```
1 <!-- edu_6_1_3.html -->
2 <!doctype html>
3 <html lang="en">
4    <head>
5        <meta charset="UTF-8">
6        <title> 设置图像宽度、高度及边框</title>
7        <style type="text/css">
8            ul{list-style-type:none; /* 去除列表项前的符号*/}
9            li{float:left;padding:0 20px; /* 垂直排列转变成水平排列 */}
10       </style>
11   </head>
12   <body>
13       <h2 align="center">设置图像宽度、高度及边框</h2>
14       <hr color="#6600cc">
15       <ul>
16           <li><img src="images1.jpg" alt="原图"></li>
17           <li><img src="images1.jpg" width="100px" alt="宽度为100px"
              border="5"></li>
18           <li><img src="images1.jpg" width="75px" height="50px" alt="
             宽75px高50像素" border="10"></li>
19       </ul>
20   </body>
21 </html>
```

视频讲解

**3. 代码解释**

代码中第 15～19 行在主体 body 标记中插入一个无序列表，并在无序列表中利用列表项插入三个图像，并对图像分“不设置高度、宽度及边框”“只设置宽度和边框”“宽度、高度及边框同时设置”等情况进行设置，并通过替代文本显示。




图 6-3　设置图像宽度和高度

## 6.1.5　设置图像对齐方式

图像和文字之间的对齐方式通过 img 标记中的 align 属性来设置。图像对齐方式分水平对齐和垂直对齐方式两种，其中水平对齐方式取值有三种：left、center、right，垂直对齐方式取值也有三种：top、middle、bottom，表示图像与同行文字的相对位置。

**1. 基本语法**

```
<img src=" URL " align="value">
```

**2. 语法说明**

align 属性的值及其说明如表 6-2 所示。

**表 6-2　align 属性的值及说明表**

| 取　　值 | 说　　明 |
|---|---|
| top | 图像的顶端和当前行的文字顶端对齐，当前行高度相应扩大 |
| middle | 图像水平中线和当前行的文字中线对齐，当前行高度相应扩大 |
| bottom | 图像的底端和当前行的文字底端对齐，当前行高度相应扩大 |
| left | 图像左对齐，浮动游离于文字之外，文字环绕图像周围，文字行高度没有任何变化 |
| center | 图像中线和当前行的文字中线对齐，当前行高度相应扩大 |
| right | 图像右对齐，浮动游离于文字之外，文字环绕图像周围，文字行高度没有任何变化 |

## 6.1.6　设置图像的间距

图像 img 标记的 hspace 和 vspace 属性用来控制图像的水平距离和垂直距离，而且两者均是以像素为单位。但在编写代码时不需要给属性值加上单位 px，否则不会产生效果。

**1. 基本语法**

```
<img src="URL"  hspace="水平间距数值"  vspace="垂直间距数值">
```

**2. 语法说明**

hspace 调整图像左右两边的空白距离，vspace 调整的是图像上下两边的空白距离。

在实际应用中很少直接使用图像的对齐属性和图像的间距属性，一般都采用 CSS 替代，所以此处不再举例。

## 6.1.7　设置图像热区链接

除了对整幅图像设置超链接外，还可以将图像划分为若干区域，这叫作“热区”，每

个区域可设置不同的超链接。此时，包含热区的图像可以称为映射图像。

### 1. 基本语法

```
<img src="图像地址" usemap="#映射图像名称">
<map name="映射图像名称">
    <area shape="热区形状" coords="热区坐标" href="URL">
</map>
```

### 2. 属性语法

usemap 属性将图像定义为客户端图像映射。图像映射指的是带有可单击区域的图像。usemap 属性与 map 标记的 name 属性相关联，usemap 属性的值以“#”开始，后面紧跟“映射图像的名称”，以建立<img>标记与<map></map>标记之间的关系。它指向特殊的<map>区域。用户计算机上的浏览器将把鼠标在图像上单击时的坐标转换成特定的行为，包括加载和显示另外一个文档。

map 标记是成对标记。name 属性映射图像的名称，与 img 标记的 usemap 属性的值关联。

area 标记是单个标记，定义图像映射中的区域。<area> 标记总是嵌套在<map></map>标记中。该标记有三个属性，分别是 shape、coords、href。href 属性定义此区域的目标 URL。shape 和 coords 属性的取值如表 6-3 所示。

**表 6-3　shape 属性与 coords 属性值对应关系及说明**

| shape 值 | 说　明 | coords 值 | 说　明 |
|---|---|---|---|
| rect | 矩形区域 | x1,y1,x2,y2 | 代表矩形两个顶点坐标 |
| circle | 圆形区域 | center-x、center-y、radius | 代表圆心和半径 |
| poly | 多边形区域 | x1,y1,x2,y2,…,xi,yi,…,xn,yn,x1,y1 | 代表各顶点坐标（首、尾坐标相同，形成封闭图形） |

【例 6-1-4】图像热区链接的应用。代码如下所示，页面效果如图 6-4 所示。

视频讲解

```
<!-- edu_6_1_4.html -->
<!doctype html>
<html lang="en">
    <head>
        <meta charset="UTF-8">
        <title>图像热区链接</title>
    </head>
    <body>
        <p>
            <a><img src="tu.jpg" align="bottom" width="200" height="150"
            border="3" alt="美女" usemap="#girl"></a>
            <map name="girl">
                <area shape="circle" href="http://www.baidu.com" coords=
                "50,50,30" alt="百度">
            </map>
        </p>
    </body>
</html>
```

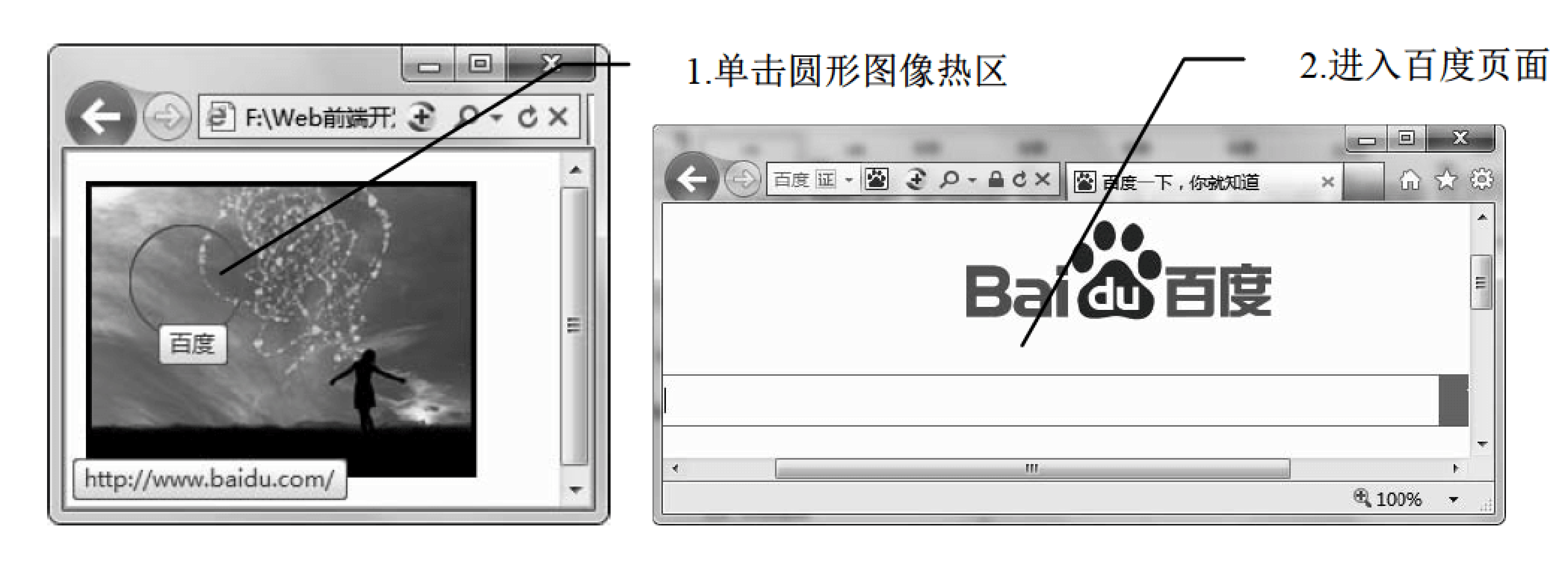



图 6-4　图像热区链接

**3. 代码解释**

代码中第 10 行定义图像链接，并在 img 标记中设置 usemap 属性引用图像热区 girl；第 11～13 行定义图像映射 map，第 12 行定义半径为 30px、圆心坐标为（50px，50px）的圆形热区，设置了热区超链接，鼠标指向热区时会显示“百度”提示信息，单击热区时会访问百度页面。

# 6.2　滚 动 文 字

要设计一个更加生动的网站，还需要在网页中添加多媒体元素。多媒体元素可以更好地体现设计者的个性，通常滚动文字可以增加文字的动态效果。

## 6.2.1　添加滚动文字

通过 marquee 标记可以添加滚动文字（内容），增加动态效果，丰富网页的内容。

**1. 基本语法**

```
<marquee width="" height="" bgcolor="" direction="up|down|left|right"
behavior="scroll|slide|alternate" hspace="" vspace="" scrollamount=""
scrolldelay="" loop="" onmouseover="this.stop()" onMouseOut="this.start()">
滚动内容</marquee>
```

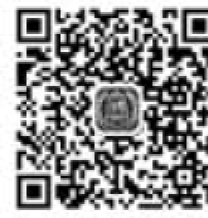
**2. 语法说明**

marquee 标记是成对标记，以<marquee>开始，以</marquee>结束，将需要滚动的内容放到 marquee 标记之间，同时也可以设置滚动内容的样式。

marquee 标记中 onMouseOver="this.stop()"属性值对的作用是当光标移动到滚动文字区域时，滚动内容将暂停滚动；onMouseOut="this.start()"属性值对的作用是当鼠标移出滚动文字区域时，滚动内容将继续滚动。

【例 6-2-1】添加滚动文字。代码如下所示，页面效果如图 6-5 所示。

```
<!-- edu_6_2_1.html -->
<!doctype html>
<html lang="en">
 <head>
  <meta charset="UTF-8">
  <title> 添加滚动文字 </title>
    <style type="text/css">
```



视频讲解

```
8        h4{font-size:20px;color:#33cc33;font-family:隶书;}
9      </style>
10  </head>
11  <body>
12  <h3 align="center">添加滚动文字</h3>
13   <hr color="#000066">
14   <marquee><h4>该文字为滚动效果</h4></marquee>
15     </body>
16   </html>
```

图 6-5　添加滚动文字

**3. 代码解释**

代码中第 14 行定义了滚动文字，文字的字体为隶书、字号为 20px、颜色为#33cc33，滚动效果为默认方式，即从右向左单向滚动。

## 6.2.2　设置滚动文字背景颜色与滚动循环

为了能够突出显示滚动的文字内容，可以通过 bgcolor 属性为滚动文字添加背景颜色，这样在网页中就会更加明显。同时也可以设置滚动的次数。

**1. 基本语法**

```
<marquee bgcolor="" loop="5" >滚动内容</marquee>
```

**2. 语法说明**

文字背景颜色采用多种方法，最常用的设置方法是十六进制和 rgb()函数。

默认情况下，滚动文字将会不停地循环滚动，但使用 loop 属性就可以设置滚动文字的滚动循环次数。循环次数直接使用数字表示。一般为整数，-1 表示无限循环。

## 6.2.3　设置滚动方向与滚动方式

在没有设定文字的滚动方向时，通常默认以从右到左的顺序滚动。在很多情况下，滚动文字可能需要从其他方向开始滚动，可以用 direction 属性进行设置。滚动文字的方向确定了以后，滚动文字就会一直滚动下去，如需要停止，则需要设置 behavior 属性来实现不同的滚动方式，如滚动一次就停止、交替滚动、循环滚动等。

**1. 基本语法**

```
<marquee direction="滚动方向" behavior="滚动方式">滚动内容</marquee>
```

**2. 语法说明**

direction 属性决定滚动方向，其取值及说明如表 6-4 所示。

表 6-4　direction 属性取值及说明

| direction 属性值 | 说明 | direction 属性值 | 说明 |
| --- | --- | --- | --- |
| up | 向上滚动 | left | 向左滚动，默认值 |
| down | 向下滚动 | right | 向右滚动 |

behavior 属性用来设置滚动方式，具体取值及说明如表 6-5 所示。

表 6-5　behavior 的属性取值及说明

| behavior 取值 | 说明 |
| --- | --- |
| scroll | 循环往复滚动，为默认值 |
| slide | 滚动一次就停止 |
| alternate | 来回交替滚动 |

## 6.2.4　设置滚动速度与滚动延迟

设置滚动文字后，可能会考虑到滚动的快慢问题， scrollamount 属性可以设置滚动文字速度。滚动延迟就是滚动文字的暂停，使用 scrolldelay 属性来设置滚动文字的延迟时间。

**1. 基本语法**

```
<marquee scrollamount="滚动速度" scrolldelay="延迟时间" >滚动内容</marquee>
```

**2. 语法说明**

滚动速度实际上就是滚动文字每次移动的长度，这个长度用数字表示，单位为像素。

延迟时间以毫秒为单位，其值设置得越小滚动速度越快。

## 6.2.5　设置滚动范围与滚动空白空间

设置滚动范围就是设置滚动的背景面积范围，在默认情况下是和文字等高、浏览器等宽的一个颜色带。该面积可以通过 width 和 height 属性来控制。

设置滚动空白空间就是指滚动文字背景和它周围的文字及图像之间的空白空间范围。默认情况下，滚动对象周围的文字或图像是与滚动背景紧密连接的，使用 hspace 和 vspace 可以设置它们之间的空白空间。

**1. 基本语法**

```
<marquee width="" height="" hspace="" vspace="" >滚动内容</marquee>
```

**2. 语法说明**

宽度值和高度值均用数字表示，单位为像素。

hspace、vspace 属性值是整数，单位为像素。

**【例 6-2-2】**滚动文字属性设置综合应用。代码如下所示，页面效果如图 6-6 所示。

```
<!-- edu_6_2_2.html -->
<!doctype html>
<html lang="en">
 <head>
  <meta charset="UTF-8">
  <title> 滚动文字属性设置综合应用 </title>
    <style type="text/css">
```

```
8         p{font-size:18px;color:#0000cc;text-indent:2em;/*首行缩进*/}
9       </style>
10      </head>
11   <body>
12    <h3 align="center">滚动文字属性设置综合应用</h3>
13      <hr color="#330099">
14      <marquee bgcolor="#c4e1c6" width="600px" height="100px" hspace="100"
      vspace="100"  direction="up" behavior="alternate" scrollamount="1"
      scrolldelay="20">
15        <p>设置滚动空白空间就是指滚动文字背景和它周围文字及图像之间的空白空间范围。默
        认情况下，滚动对象周围的文字或图像是与滚动背景紧密连接的，使用hspace和vspace
        可以设置它们之间的空白空间。</p>
16      </marquee>
17    </body>
18  </html>
```

视频讲解

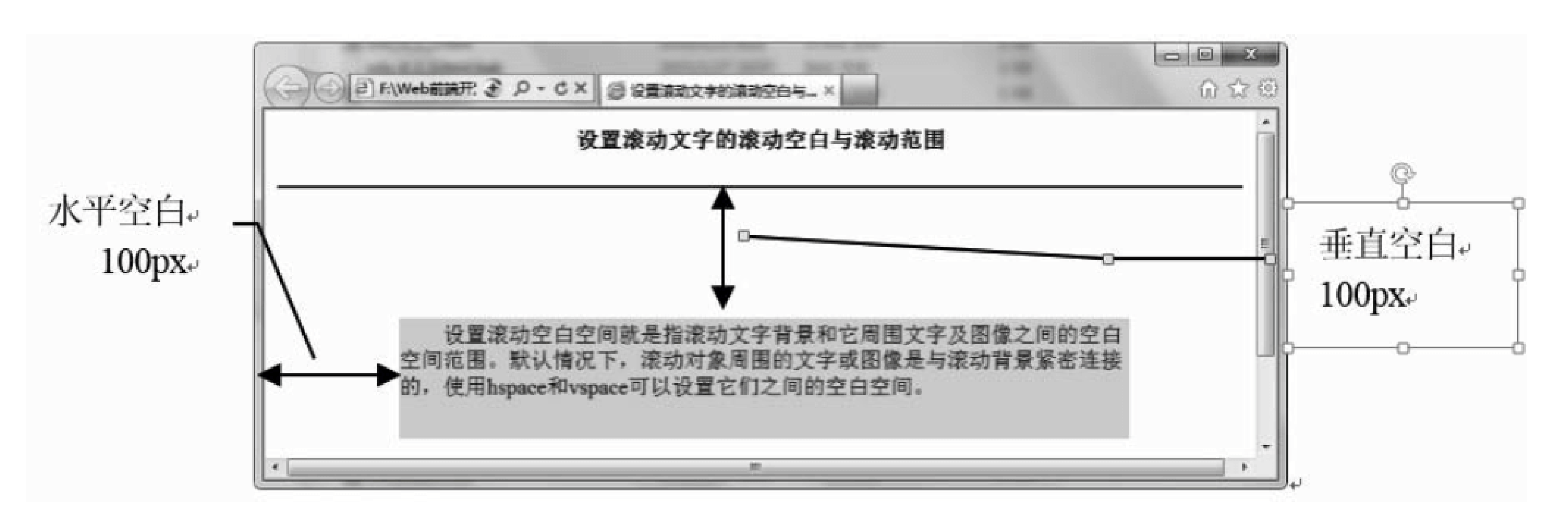



图 6-6　滚动文字属性设置综合应用

**3. 代码解释**

代码中第 14 行定义了滚动文字的背景颜色为#c4e1c6、宽度为 600px、高度为 100px、背景与周边元素水平空间空白为 100px，背景与周边元素垂直空白空间为 100px，滚动方向为向上、滚动行为为交替滚动、滚动速度为 1px、滚动时延为 20ms。

## 6.3　音频、视频及 Flash 文件

除了滚动文字外，网页中的多媒体文件还包括音频文件、视频文件以及 Flash 文件，可以为网页增加背景音乐等效果。

使用<embed></embed>标记，可以播放的文件类型有 Midi、Mav、AIFF、SWF、AV、MP3、MOV、AVI 等。

**1. 基本语法**

```
<embed src="多媒体文件" width="界面的宽度" height="界面的高度"autostart=
"true|false" loop=" true|false "></embed>
```

**2. 语法说明**

- width、height：整型值，单位为像素。设置宽度和高度会出现播放界面，否则不显示播放界面。一些高版本浏览器不设置宽度和高度也可以出现播放界面。如果播放声音、音乐文件作为背景音乐时，必须同时将宽度和高度属性的值设置为 0。
- src：设置媒体文件的路径。

- autostart：逻辑值。true 为自动播放；false 为不自动播放。
- loop：逻辑值。规定音频或视频文件是否循环。属性值为 true 时，音频或视频文件循环；属性值为 false 时，音频或视频文件不循环。

【例 6-3-1】音视频及 Flash 文件播放。代码如下所示，页面效果如图 6-7 所示。

视频讲解

```
1 <!-- edu_6_3_1.html -->
2 <!doctype html>
3 <html lang="en">
4    <head>
5        <meta charset="UTF-8">
6        <title>页面中嵌入多媒体文件</title>
7        <style type="text/css">
8            div{text-align:center;font-size:18px;font-family:黑体;}
9        </style>
10   </head>
11   <body>
12       <div id="" class="">
13           <h3>醉花阴</h3>
14           <h4>李清照</h4>
15           <hr size="5" color="#660099">
16           <p>薄雾浓云愁永昼，瑞脑消金兽。<br>佳节又重阳，玉枕纱厨，半夜凉初透。
             <br> 东篱把酒黄昏后，有暗香盈袖。<br>莫道不消魂，帘卷西风，人比黄花瘦。</p>
17           <hr size="5" color="#660066">
18           <h3>嵌入的多媒体文件</h3>
19           <!-- <bgsound src="embed/53.mid" loop="-1"> -->
20           <embed src="蔡琴明月几时有.mp3" width="300" height="150"
             autostart="true" loop="true" ></embed>
21           <embed src="093zhy.swf" width="300" height="150" autostart=
             "true" loop="true"></embed>
22       </div>
23   </body>
24 </html>
```

### 3. 代码解释

代码中第 20 行通过 embed 标记嵌入了 mp3 文件。第 21 行通过 embed 标记嵌入了一个 Flash 文件。




图 6-7　插入音视频及 Flash 文件

## 6.4 综合实例

以“杭州华三通信技术有限公司”的主网站为例，运用图像、视频及背景音乐等多媒体元素来设计一个简化的 H3C 页面，如图 6-8 所示。

视频讲解




图 6-8　图像与多媒体文件应用

```
<!--edu_6_4_1.html -->
<!doctype html>
<html lang="en">
   <head>
      <meta charset="UTF-8">
      <title>图像与多媒体文件应用</title>
      <style type="text/css">
      ul{list-style-type:none;}
      li{display:inline;margin:0px 10px;}
      marquee{clear:both;}
      p{text-indent:2em;}
      #div1{background:#99ffcc;height:60px;padding:10px50px;margin:0 auto;}
      img{float:left;margin-left:50px;}
      #ul1{float:left;padding-top:25px;padding-left:20px;}
      #ul1 li{width:100px;}
      #div2{background:#00cc00;height:500px;}
      </style>
   </head>
   <body>
      <div id="div1" class="">
      <img src="h3clogo.png" alt="">
      <ul id="ul1">
         <li><a href="">产品技术</a></li>
         <li><a href="">解决方案</a></li>
         <li><a href="">服务支持</a></li>
         <li><a href="">培训认证</a></li>
```

```
27            <li><a href="">合作伙伴</a></li>
28            <li><a href="">关于我们</a></li>
29        </ul>
30        </div>
31        <div id="div2" class="">
32            <!--<bgsound src="exam01.mp3" >  -->
33            <ul>
34                <li><img src="h3ccloud.jpg" width="300" height="230"
               border="0" alt=""></li>
35                <li><embed src="h3c_newit1.swf" loop="true" autostart=
               "true" width="400" height="300"></embed></li>
36            </ul>
37            <marquee behavior="alternate" direction="up" height="100px"
           scrolldelay="500" bgcolor="#ffffff">
38            <p>云彩虹（Cloud Rainbow）的解决方案，可以实现在上、下级两级云资源管
           理平台的备份、资源弹性扩展与业务迁移分发，打破IT资源与业务只能本地部署
           的局限性，统一企业多级云资源的部署和管理，通过跨区域的备份与迁移提升业务
           的连续性，为企业提供全局性的IT资源管理视角。</p>
39            </marquee>
40            <hr color="red">
41            <p align="center">杭州华三通信技术有限公司.保留一切权利.浙ICP备
           09064986号</p>
42        </div>
43    </body>
44 </html>
```

上述代码中第 20～30 行在第 1 个 div 中插入一个 H3C 公司的 logo 和一个导航菜单；第 31～42 行在第 2 个 div 中分别插入背景音乐、图像、Flash、滚动文字等；第 8～16 行分别定义 ul、li、marquee、img、p 等标记样式及#div1、#div2、#ul1 等 id 样式。

# 本 章 小 结

本章主要介绍了在网页中插入图像、滚动文字、音频及其他多媒体文件的方法。着重讲授了 img 标记、marquee 标记、embed 标记的语法及其属性的设置方法。

运用这些标记可以对所开发的网站进行重新布局、页面美化，不断改善用户体验，吸引更多网络访问者浏览自己的网站。

# 练习与实验

## 练习 6

**1. 选择题**

（1）指定滚动文字的滚动延时正确的标记是（　　）。

（A）<marquee scrollamount ="200"> ... </marquee>

（B）<marquee loop="200"> ... </marquee>

（C）<marquee auto="200"> ... </marquee>

（D）<marquee scrolldelay="200"> ... </marquee>

（2）能够播放 Flash 和视频文件的 HTML 标记是（　　）。

（A）<embed src=""></embed>　　（B）<bgsound src="" />

（C）<marquee></marquee>　　（D）<a href=""></a>

（3）<img alt="这是图像">，这个标记作用是（　　）。

（A）添加图像链接

（B）决定图像的排列方式

（C）在浏览器完全读入图像时，在图像位置显示的文字

（D）在浏览器尚未完全读入图像时，在图像的上方显示的“×”，并显示替代文本

（4）HTML 代码<a href="#"><img src="/blog/name/images.jpg"></a>表示（　　）。

（A）按某种方式对齐加载的图像　　（B）设置一个图像链接

（C）设置围绕一个图像的边框的大小　　（D）加入一条水平线

**2. 填空题**

（1）网页中插入图像使用________标记，插入多媒体文件使用________________标记，插入滚动文字使用___________________标记。

（2）在给图像指定超链接时，默认情况下总是会显示蓝色边框，如果不想显示蓝色边框，应使用以下语句：<a href="test.html"><img src="image1.gif" ________________></a>。

（3）热区 area 标记的 shape 属性取值为"rect"表示热区的形状为______；shape 属性取值为"circle"表示热区的形状为______；shape 属性取值为"poly"表示热区的形状为______。

**3．简答题**

（1）设置滚动文字 marquee 标记的 hspace 和 vspace 属性的作用是什么？

（2）使用<img>标记可以在页面中插入图像，如何设置图像的高度和宽度？如何设置替换文本？

# 实验 6

1. 设置图像的相关对齐属性，编程实现如图 6-9 所示的效果。

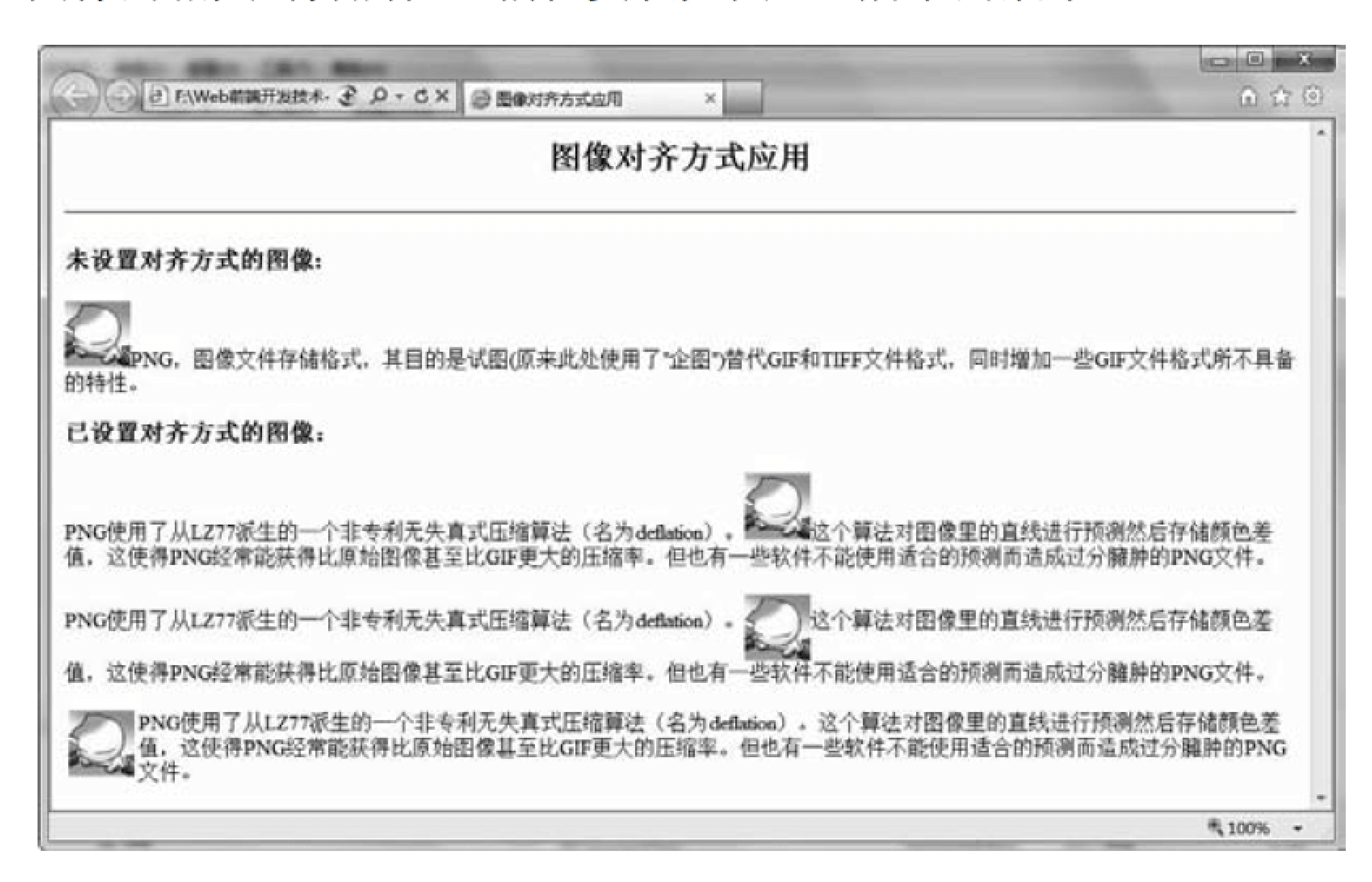



图 6-9　图像对齐方式应用

2. 设计一个图像画廊，页面效果如图 6-10 所示。采用无序列表加载五幅图像，并利用滚动文字 marquee 标记及其属性的设置实现五幅图像从右向左交替滚动显示。

设计中需要用到样式表（直接将下述代码插入到头部 head 标记中）如下所示：

```
<style type="text/css">
    img{width:100px;height:100px;border:2px #cc0066 ridge;}
    ul{list-style-type:none;}
    li{float:left;}
</style>
```




图 6-10　图像画廊

# 第 7 章　CSS 基础

本章学习目标

**在网页设计过程中经常会遇到需要对网页中的同样的内容进行重复的属性设置，这既浪费时间，也增加了代码冗余，还带来了后期网站改版维护困难等诸多问题，CSS（Cascading Style Sheet，层叠样式表）就是为了简化页面元素修饰、美化工作而诞生的。所以本章引入 CSS，主要是为了实现对网页的字体、颜色、布局等元素进行精确控制，解决网页内容与网页表现分离的问题，进一步提高网站的可维护性，方便网站快速重构，实现网站定期换肤的功能。**

**Web 前端开发工程师应掌握以下内容：**

- **理解 CSS 的概念、特点。**
- **掌握 CSS 基本语法、选择器分类与声明的结构。**
- **掌握 CSS 的定义及引用的方式。**
- **理解 CSS 继承与层叠的含义。**

## 7.1　CSS 概念

CSS 属于动态 HTML 技术，它扩充了 HTML 标记的属性设置，使得页面显示效果更加丰富，表现效果更加灵活，它与 div 的配合使用可以很好地对页面进行分割和布局。传统 HTML 网页设计往往是内容和表现混合，随着网站规模不断扩大，无论是修改网页还是维护网站都显得越来越困难。CSS 对页面元素、布局等能够更加精确控制，同时能够实现内容和表现的分离，使得网站的设计风格趋向统一、维护更加容易。

### 7.1.1　CSS 的基本概念

CSS 也称为级联样式表，用来进行网页风格设计。由 Hakon Wium Lie（哈肯 •维姆 •莱，挪威）和 Bert Bos（伯特 • 波斯，荷兰）于 1994 年共同发明。

在网页设计时采用 CSS 技术，可以有效地对页面的布局、字体、颜色、背景和其他效果实现更加精确的控制。只要对相应的代码做一些简单的修改，既可以改变同一页面的不同部分效果，也可以改变同一个网站中不同网页的外观和格式。

### 7.1.2　传统 HTML 的缺点

HTML 标记是用来定义文档内容，例如通过 h1、p、table 等标记表达“这是标题”“这是段落”“这是表格”等信息，而文档布局由浏览器完成。随着新的 HTML 标记（例如字

体标记和颜色属性）添加到 HTML 规范中，要实现页面美工、文档内容清晰、独立于文档表现层的站点变得越来越困难。传统 HTML 的缺点主要体现在如下几个方面。

**1. 维护困难**

为了修改某个特殊标记的格式，需要花费很多的时间，尤其对于整个网站而言，后期修改和维护的成本很高。

**2. 标记不足**

HTML 自身的标记并不丰富，很多标记都是为网页内容服务的，而关于美工的标记，例如文字间距、段落缩进等，在 HTML 中都很难找到。

**3. 网页过“胖”**

由于对各种风格样式没有统一进行控制，用 HTML 编写的页面往往体积过大，占用了很多宝贵的带宽。

**4. 定位困难**

在整体页面布局时，HTML 对于各个模块的位置调整显得捉襟见肘。

## 7.1.3 CSS 的特点

CSS 通过定义标记或标记属性的外在表现，对页面结构风格进行控制，分离文档的内容和表现，克服了传统 HTML 的缺点。将 CSS 嵌入在页面中，通过浏览器解释执行，而且 CSS 文件是文本文件，只要理解了 HTML 就可以掌握它。

## 7.1.4 CSS 的优势

CSS 可以称得上 Web 设计领域的一个突破，它的诞生使得网站开发者如鱼得水，其具有以下几点优势。

**1. 表现和内容分离**

CSS 通过定义 HTML 标记如何显示控制网页的格式，使得页面内容和表现分离，简化了网页格式设计，也使得对网页格式的修改更方便。

**2. 增强了网页的表现力**

CSS 样式属性提供了比 HTML 更多的格式设计功能。例如，可以通过 CSS 样式去掉网页中超级链接的下画线，甚至可以为文字添加阴影、翻转效果等。

**3. 使整个网站显示风格趋于统一**

将 CSS 样式定义到样式表文件中，然后在多个网页中同时应用样式表文件中的样式，就可以确保多个网页具有一致的格式，并且可以随时更新样式表文件，实现自动更新多个网页的格式功能，从而大大降低了网站的开发与维护的成本。

## 7.1.5 CSS 的编辑方法

编辑 CSS 主要有两种方式：

（1）写在 HTML 文件里面的 CSS 规则。根据其位置又可以分为两种形式：一种是写在某个元素的属性部分，作为 style 属性的值；另一种是写在 head 标记里面，通过 style 标记包含。

（2）将 CSS 规则写在单独的文件里。建议采用此种方式，该文件称为 CSS 文件，它

是纯文本文件，可以使用任何编辑器编辑，文件后缀名为.css。在需要应用 CSS 规则的多个 HTML 文件里面引用该 CSS 文件，从而实现内容和表现的分离，也提高了网站可维护性。

# 7.2　使用 CSS 控制 Web 页面

CSS 控制页面是通过 CSS 规则实现的，CSS 规则由选择器和声明组成，声明由属性和属性值对组成。CSS 提供了丰富的选择器类型，包括标记选择器、类选择器、id 选择器、伪类选择器及属性选择器等，能够灵活地对整个页面、页面中的某个标记或一类标记进行样式设置。

## 7.2.1　CSS 基本语法

CSS 就是一个包含一个或多个规则的文本文件。CSS 规则由两个主要的部分构成：选择器（Selector）和声明（Declaration）。

选择器通常是需要改变样式的 HTML 元素。

声明由一个或多个属性与属性值对组成。属性是 CSS 的关键字，如 font-family（字体）、color（颜色）和 border（边框）等。属性用于指定选择器某一方面的特性，而属性值则用于指定选择器的特性的具体特征。

**1. 基本语法**

```
selector {property1: value1; property2: value2; property3: value3;…}
```

**2. 语法说明**

1）选择器

选择器可以是 HTML 标记名称或者属性的值，也可以是自定义的标识符。

2）属性/属性值对

“属性:属性值”必须一一对应，属性与属性值之间必须用“:”连接，每个属性/属性值对之间用分号(;)分隔。

3）属性

在 CSS 中对属性命名与脚本语言中对属性命名有一点不同，即属性名称的写法，在 CSS 中，凡是属性名为两个或两个以上的单词构成时，单词之间以连词符号(-)分隔，例如背景颜色属性 background-color；而在脚本中，对象属性则连写成 backgroundColor，如果属性由两个以上单词构成，则从第二个单词开始向后，所有单词首字母必须大写。

下面是一个简单样式表的示例：

```
p{background-color:red; font-size:20px; color:green; }
```

上例的 CSS 规则中 p 为选择器，background-color、font-size、color 为属性，red、20px、green 为属性值，该 CSS 规则将 HTML 中的所有段落统一设置成“背景色为红色、字体大小为 20px 以及字体颜色为绿色”。通常为了增强样式定义的可读性，建议每行只描述一个属性，格式如下所示：

```
p{
    background-color:red;
    font-size:20px;
    color:green;
}
```

4）复合属性

在 CSS 中，有些属性可以表示多个属性的值。如关于文字的设置，有 font-family、font-size、font-style，这些可以用一个属性 font 来表示，例如：

```
p{ font-style:italic; font-size:20px; font-family: 黑体; }
```

可以直接使用 p{ font:italic 20px 黑体; }来表示。

值得注意的是，使用 font 复合属性在一个声明中设置所有字体属性时，应按照 font-style、font-variant、font-weight、font-size/line-height、font-family 的顺序，可以不设置其中的某个值，例如“font:100% verdana;”仅设置了 font-size、font-family 属性，其他未设置的属性会使用其默认值。类似的复合属性还有 border、margin、padding 等。

5）多个属性值

在 CSS 中，有些属性可以设置多个属性值，用逗号(,)分隔。例如：

```
selector{ font-family: "楷体_gb2312 ", "黑体", " Times New Roman "; }
```

该样式表说明了可以使用楷体_gb2312、黑体、Times New Roman 三种字体来设置 selector 的字体效果。若系统中找不到楷体_gb2312，则使用黑体；若也没有黑体，则使用 Times New Roman，即按出现的先后顺序优先选择。

6）CSS 注释

像其他语言一样，CSS 允许用户在源代码中嵌入注释。CSS 注释被浏览器忽略，不影响网页效果。注释有助于记住复杂的样式规则的作用、应用的范围等，便于样式规则的后期维护和应用。CSS 注释以字符“/*”开始，以字符“*/”结束。下面是注释样例：

```
/*  这是多行注释   CSS文件名: out.css
    功能说明：定义样式
*/
/* 单行注释    样式  段落p */
p{font-size:20px; /* 行尾注释   定义字号*/ }
```

“/* … */”这种格式可以单独一行书写，也可以写在语句的后面，可以注释一行，也可以注释多行。注释不能嵌套。

### 7.2.2　CSS 选择器类型

CSS 选择器主要有五种类型：标记选择器、类选择器、id 选择器、伪类选择器及属性选择器等。

**1. 标记选择器**

标记选择器（也可称为“元素选择器”）即直接使用 HTML 标记名称作为选择器，它定义的样式作用于页面中所有与选择器同名的标记，前面的示例代码均属于标记选择器，这里不再详细介绍。

**2．类选择器**

任何合法的 HTML 标记都可以使用 class 属性，class 属性用于定义页面上的 HTML 元素标记组，这些标记组通常具有相同的功能或作用，因此它们可以设置相同的样式规则。

首先创建类，用户需要给它命名，类名可以是任何形式，建议读者以描述性的名称来起名，这样对于整个代码的维护及协同开发有很大帮助。为类选择了名字之后，用户可以通过设置 class 属性为 HTML 标记分配类。如果是多个类要用空格分隔，那么 HTML 标记可以是多个类的一部分。示例代码如下所示：

```
<p class="c2">著名诗人</p>
<ol class="c1">
    <li class="c2">李白</li>
    <li class="c3 c4">杜甫</li>
    <li>杜牧</li>
</ol>
```

在 HTML 标记中设置了 class 属性之后，用户可以使用它作为 CSS 的类选择器。

类选择器由点号“.”及类名称直接相连构成。示例代码如下所示：

```
.c2{ color:red; font-weight:bold; }
.c3{ font-style:italic; }
```

标记选择器和类选择器可以联合使用，使用方式是标记选择器与类选择器直接相连，称为**联合选择器**，可以用来设置特定类中的特定标记。示例代码如下所示：

```
p.c2{ color:green; font-size:20px; }
li.c3{ color:red; }
```

在上面的代码中，前者选择所有 class="c2"的<p>元素，后者选择所有 class="c3"的<li>元素。

**3．id 选择器**

HTML 标记的 id 属性与 class 属性类似，可以用于各类标记中，也可以作为 CSS 选择器来使用。id 具有很多限制，只有页面上的标记（body 标记及其子标记）才能具有给定的 id。在 HTML 文件内，每个 id 属性的取值必须唯一，且只能用于指定的一个标记。id 属性的取值必须以字母开头，由字母、数字、下画线、连字符组成。如果作为 CSS 选择器使用，通常建议使用字母和数字及下画线组合作为 id 名称。

id 选择器由井号“#”及 id 属性值直接相连构成。示例代码如下所示：

```
#right{ color:red; text-align:right; font-size:20px; }
<p id="right">使用id选择器设置样式。</p>
```

对于 CSS 来说，id 选择器与 class 选择器的功能很相似但不完全相同。一般来说，class 选择器更加灵活，能完成 id 选择器的所有作用，也能完成更加复杂的功能应用。如果对样式可重用性要求较高，则应该使用 class 选择器将新元素添加到类中来完成。对于需要唯一标识的页面元素，则可以使用 id 选择器。

**4．伪类选择器**

前面介绍的选择器都是能够与 HTML 中具体标记对应的，但是像段落的第 1 行、超链接访问前与访问后等，就没有 HTML 标记与之对应，从而也没有简单的 CSS 选择器应用，

为此 CSS 引进了伪类选择器。其用法如下：

```
标记:伪类名{ /* CSS规则 */ }
```

常用伪类如表 7-1 所示。

表 7-1　常用伪类

| 伪　类　名 | 说　　明 |
|---|---|
| link | 设置 a 标记在未被访问前的样式 |
| hover | 设置 a 标记在鼠标悬停时的样式 |
| active | 设置 a 标记在被用户激活（在鼠标点击与释放之间）时的样式 |
| visited | 设置 a 标记在被访问后的样式 |
| first-letter | 作用于块，设置第一个字符的样式 |
| first-line | 作用于块，设置第一个行的样式表 |
| first-child | 设置第一个子标记的样式 |
| lang | 设置具有 lang 属性的标记的样式 |

【例 7-2-1】伪类选择器演示。代码如下所示，页面效果如图 7-1 所示。

视频讲解

```
<!-- edu_7_2_1.html -->
<!doctype html>
<html lang="en">
    <head>
        <meta charset="UTF-8">
        <title>选择器演示</title>
        <style type="text/css">
            a:link{color:gray;text-decoration:none;}
            a:visited{color:blue;text-decoration:none;}
            a:hover{color:red;text-decoration:underline;}
            a:active{color:yellow;text-decoration:underline;}
            p:first-letter{font-weight:bold;font-family:"黑体";}
            p:first-line{font-size:32px;}
        </style>
    </head>
    <body>
        <p>在支持CSS的浏览器中，链接的不同状态都可以不同的方式显示，这些状态包括：活动状态，已被访问状态，未被访问状态和鼠标悬停状态。<br>
        注意：a:hover 必须被置于a:link 和a:visited 之后，才是有效的。a:active必须被置于a:hover之后，才是有效的。
        </p>
        <a href="http://www.baidu.com">搜索一下：百度</a>
    </body>
</html>
```




图 7-1　伪类选择器

### 5. 代码解释

代码中第 8～13 行定义伪类选择器，分别设置了超链接未访问、已访问、鼠标悬停、激活的样式以及段落第 1 个字黑体加粗、第 1 行字号 32px 的样式。

特别应注意 a:hover 必须置于 a:link 和 a:visited 之后，才是有效的；a:active 必须置于 a:hover 之后，才是有效的。设置的顺序如下：

```
a:link{color:blue;}
a:visited{color:blue;}
a:hover{color:red;}
a:active{color:yellow;}
```

### 6. CSS 属性选择器

除了使用 CSS 的标记、class、id、伪类选择器外，还可以使用属性选择器给带有指定属性的 HTML 标记设置样式，如表 7-2 所示。低版本的浏览器不支持属性选择器。

表 7-2　CSS 属性选择器及描述表

| 选　择　器 | 描　　述 |
|---|---|
| [attribute] | 用于选取带有指定属性的标记 |
| [attribute=value] | 用于选取带有指定属性和值的标记 |
| [attribute~=value] | 用于选取属性值中包含指定词汇的标记（用空格分隔的字词列表） |
| [attribute\|=value] | 用于选取带有以指定值开头的属性值的标记（属性值是 value 或者以“value-”开头的值） |
| [attribute^=value] | 匹配属性值以指定值开头的每个标记 |
| [attribute$=value] | 匹配属性值以指定值结尾的每个标记 |
| [attribute*=value] | 匹配属性值中包含指定值的每个标记 |

1）属性选择器

定义属性选择器时，需要通过方括号“[]”将属性包围住，例如[target]、[color]。只需要匹配属性名。格式如下所示：

```
[属性名]{属性:属性值;属性:属性值;...;}
[title]{color:red;} /*  带有title属性的所有元素设置样式:  */
```

2）属性和值选择器

指定属性名，同时指定了该属性的属性值，以指定“属性/值”的所有标记设置样式。例如为[class="p1"]的所有段落设置统一样式。格式如下所示：

```
[class="p1"]{font-size:24px;color:red;border:5px solid blue;}
```

3）多个值的属性和值选择器

可以对具有指定值的 name 属性的所有标记设置样式。适用于由空格分隔的属性值。

```
[name~=value]{background:#FF00CC;}  /*属性值是以空格分隔的词汇列表中一个单独的词*/
[name^=value]{background:#FF00CC;} /*属性值是以value开头的*/
[name$=value]{background:#FF00CC;} /*属性值是以value结尾的*/
[name*=value]{background:#C3C;}     /*属性值中包含了value*/
[name|=value]{background:#C3C;}     /*属性值是value或者以"value-"开头的值*/
```

【例 7-2-2】属性选择器的应用。代码如下所示，页面效果如图 7-2 所示。

视频讲解

```
<!-- edu_7_2_2.html -->
<!doctype html>
<html lang="en">
    <head>
        <meta charset="UTF-8">
        <title>属性选择器的应用</title>
        <style type="text/css">
            [title]{font-size:18px;color:green;}
            p[name="chu"]{font-style:italic;}
            p[name~="chu"]{font-weight:bold;}
            p[name^="chu"]{text-align:center;}
            p[name$="jiu"]{color:blue;}
            p[name*="jiang"]{color:red;text-decoration:underline;}
        </style>
    </head>
    <body>
        <h3>属性选择器的应用</h3>
        <p title="p1" name="chu"> [title][name="chu"]属性和值选择器,绿色、18px、斜体、居中</p>
        <p name="jiu chu ">[name="jiu chu "]属性值包含指定值的选择器，标粗</p>
        <p name="linchujiu">属性值中以jiu结尾的，蓝色</p>
        <p name="changjianghuanghe">属性值中包含jiang字符串，红色、下画线</p>
    </body>
</html>
```




图 7-2　属性选择器综合应用（Chrome 浏览器）

## 7.2.3　CSS 选择器声明

在声明各种 CSS 选择器时，如果某些选择器的风格是完全相同的，或者部分相同，都可以利用集体声明的方法，用“,”分隔多个选择器，对风格相同的 CSS 选择器同时声明。

### 1. 集体声明

集体声明示例代码如下：

```
h1,h2,h3,h4,h5,p{ color:purple; font-size:16px; }
h2.special, .special, #one{ text-decoration:underline;}
```

### 2. 全局声明

对于实际网站中的一些小型页面，例如弹出的小对话框和上传附件的小窗口等，希望

这些页面中所有的标记都使用同一种 CSS 样式，但又不希望通过逐个加入集体声明列表的方式，这时可以利用全局声明符号“*”。示例代码如下：

```
*{ color:purple; font-size:16px; }
```

**3. 派生选择器（上下文选择器）**

另外，根据标记所在位置的上下文关系来定义样式，可以使标记更加简洁。派生选择器允许根据文档的上下文关系来确定某个标记的样式。通过合理地使用派生选择器，可以使 CSS 代码变得更加整洁。

例如,要让列表项<li>中的<strong>标记变为斜体字，而不是通常的粗体字，可以这样定义一个派生选择器：

```
li strong { font-style: italic; font-weight: normal; }
strong{ font-weight:bold; }
```

测试代码如下：

```
<p><strong>我是粗体字，不是斜体字，因为我不在列表当中，所以这个规则对我不起作用
</strong></p>
<ol>
  <li><strong>我是斜体字。这是因为strong元素位于li标记内。</strong></li>
   <li>我是正常的字体。</li>
</ol>
```

在上面的例子中，有两个 strong 标记，但只有 li 元素中的 strong 元素的样式为斜体字，而且无须为 strong 标记定义特别的 class 或 id，应用派生选择器，代码更加简洁。

## 7.2.4　CSS 定义与引用

CSS 按其位置可以分为四种：内联样式表（Inline Style Sheet）、内部样式表（Internal Style Sheet）、链接外部样式表（Link External Style Sheet）以及导入外部样式表（Import External Style Sheet）。

**1. 内联样式表（行内样式表）**

内联样式表的 CSS 规则写在首标记内，只对所在的标记有效。几乎任何一个 HTML 标记上都可以设置 style 属性。属性值可以包含 CSS 规则的声明，不包含选择器。

1）基本语法

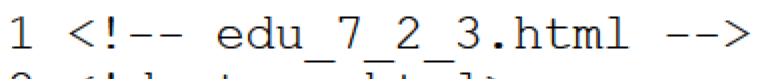
```
<标记 style="属性1：属性值1;属性2：属性值2;...">修饰的内容</标记>
```

2）语法说明

- 标记是指 HTML 标记，如 p、h1、body 等标记。
- 标记的 style 定义的声明只对自身起作用。
- style 属性的值可以包含多个属性/值对，每一属性/值对之间用“;”分隔。
- 标记自身定义的 style 样式优先于其他所有样式定义。

【例 7-2-3】内联样式表的使用。代码如下所示，页面效果如图 7-3 所示。


视频讲解



```
<!-- edu_7_2_3.html -->
```



```
<!doctype html>
```



```
<html lang="en">
```

```
4      <head>
5          <meta charset="UTF-8">
6          <title>内联样式（Inline Style）</title>
7      </head>
8      <body>
9          <p style="font-size:20px;font-style:italic;">这个内联样式（Inline
           Style）定义段落文字大小20px，文字风格为斜体。</p>
10         <p>这段文字没有使用内联样式。</p>
11     </body>
12 </html>
```

3）代码解释

代码中第 9 行采用内联样式表定义段落 p 标记的样式。通过设置 style 属性值为“font-size:20px;font-style:italic;”来实现。style 属性的值相当于 CSS 规则中声明部分，由多个“属性/值”对构成，“属性/值”对之间用“;”分隔。

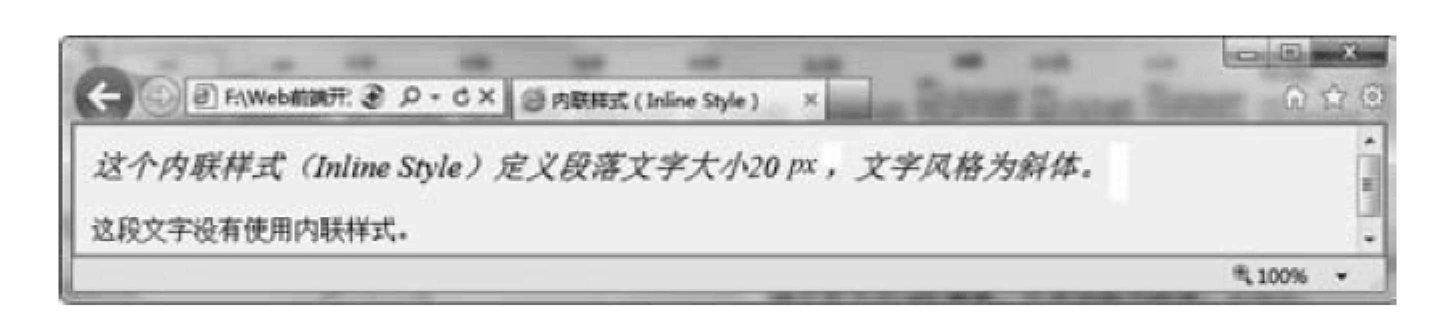



图 7-3　内联样式表

值得注意的是，内容和表现的分离是创建 CSS 的初衷，这一技术的产生将使内联样式的应用大为减色，使用 HTML+CSS 的方式更有意义。除非有特别的用途，否则开发者应该避免使用内联样式表。

**2. 内部样式表**

内部样式表写在 HTML 的<head></head>里面，只对所在的网页有效。使用<style></style>标记对来放置 CSS 规则。

1）基本语法

```
<style type="text/css">
    选择器1{属性1：属性值1;属性2：属性值2;...}
    选择器2{属性1：属性值1;属性2：属性值2;...}
    ...
    选择器n{属性1：属性值1;属性2：属性值2;...}
</style>
```

2）语法说明

- style 标记是成对标记，有一个 type 属性是指 style 元素以 CSS 的语法定义。
- 选择器 1、……、选择器 n，可以定义 n 个选择器，再定义声明部分。
- 属性和属性值之间用冒号连接，“属性/属性值”对之间用分号分隔。

【例 7-2-4】内部样式表的使用。代码如下所示，页面效果如图 7-4 所示。



视频讲解

```
<!-- edu_7_2_4.html -->
<!doctype html>
<html lang="en">
    <head>
        <meta charset="UTF-8">
        <title>内部样式（Internal Style）</title>
        <style type="text/css">
```

```
8           .int_css{
9               border-width:2px;        /*定义边框宽度*/
10              border-style:solid;      /*定义边框样式*/
11              text-align:center;       /*定义文本对齐方式*/
12              color:red;               /*定义颜色*/
13          }
14          #h1_css{
15              font-size:28px;          /*定义字体大小*/
16              font-style:italic;       /*定义字体样式*/
17          }
18      </style>
19  </head>
20  <body>
21      <h1 class="int_css">h1这个标题使用类样式。</h1>
22      <h1 id="h1_css">h1这个标题使用ID样式。</h1>
23      <h1>h1这个标题没有使用样式。</h1>
24  </body>
25 </html>
```

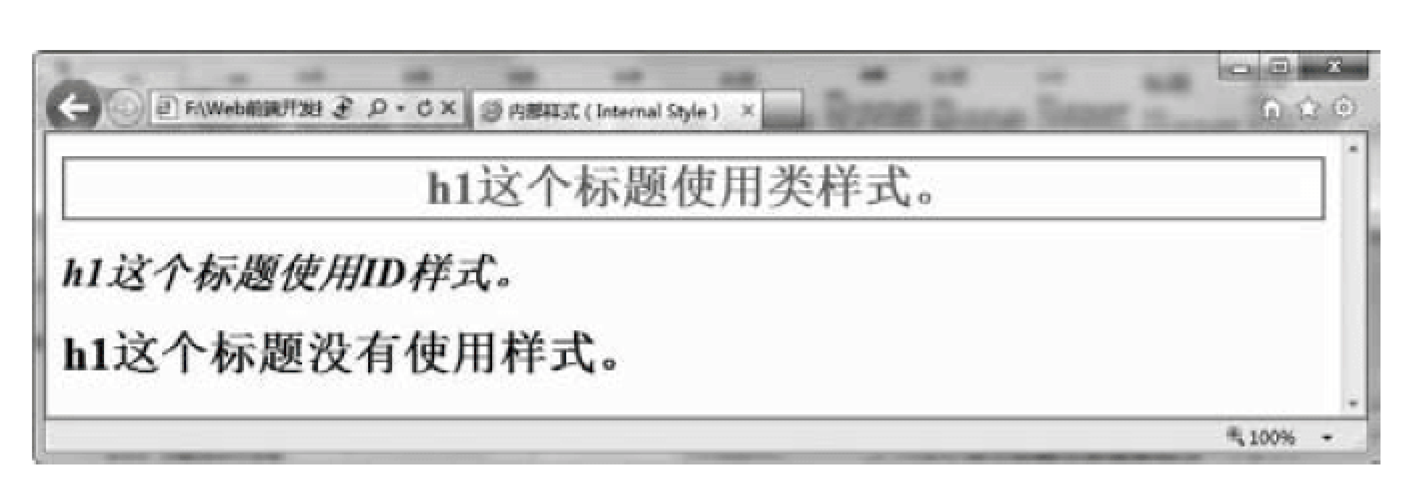



图 7-4　内部样式表

3）代码解释

代码中第 7 行～第 18 行是定义内部样式表。其中第 8 行定义类选择器；第 14 行定义 id 选择器；第 21 行引用类样式，只有设置了 class 属性值为 int_css 的标记规则才会生效。第 22 行引用 ID 样式，只有设置了 id 属性值为 h1_css 的标记规则才会生效。第 23 行是默认样式。

**3. 外部样式表**

外部样式表是将 CSS 规则写在以.css 为后缀的 CSS 文件里，在需要用到此样式的网页里引用该 CSS 文件。一个 CSS 文件可以供多个网页引用，从而实现整体页面风格统一设置。根据引用的方式不同可以分为链接外部样式表和导入外部样式表，它们形式上的区别在于链接外部样式表通过链接 link 标记，导入外部样式表必须在内部样式表内首行通过“@import url（"外部样式文件名称"）;”来定义。

1）链接外部样式表

（1）基本语法。

```
<link type="text/css" rel="stylesheet" href="out.css">
```

（2）语法说明。

link 标记是单个标记，也是空标记，它仅包含属性。此标记只能存在于 head 部分，不过它可出现任何次数。link 标记的属性、取值及说明如表 7-3 所示。

表 7-3　link 标记的属性、取值及说明

| 属　　性 | 取　　值 | 说　　明 |
|---|---|---|
| type | MIME_type | 规定被链接文档的 MIME 类型 |
| rel | stylesheet | 定义当前文档与被链接文档之间的关系 |
| href | URL | 定义被链接文档的位置 |

【例 7-2-5】链接外部样式表的使用。代码如下所示，页面效果如图 7-5 所示。



视频讲解

- CSS 文件 out.css

```
/*样式表文件 out.css*/
.int_css{
    border-width:2px;       /*定义边框宽度*/
    border-style:solid;     /*定义边框样式*/
    text-align:center;      /*定义文本对齐方式*/
    color:green;            /*定义颜色*/
}
#h1_css{
    font-size:28px;         /*定义字体大小*/
    font-weight:bold;       /*定义字体粗细*/
}
```

- HTML 文件 edu_7_2_5.html

```
1 <!-- edu_7_2_5.html -->
2 <!doctype html>
3 <html lang="en">
4     <head>
5         <meta charset="UTF-8">
6         <title>链接外部样式（External Style）</title>
7         <link type="text/css" rel="stylesheet" href="out.css">
8     </head>
9     <body>
10        <h1 class="int_css">这个标题h1使用了链入外部样式中的类样式。</h1>
11        <h1 id="h1_css">这个标题h1使用链入外部样式中的ID样式。</h1>
12        <h1>这个标题h1没有使用样式。</h1>
13    </body>
14 </html>
```

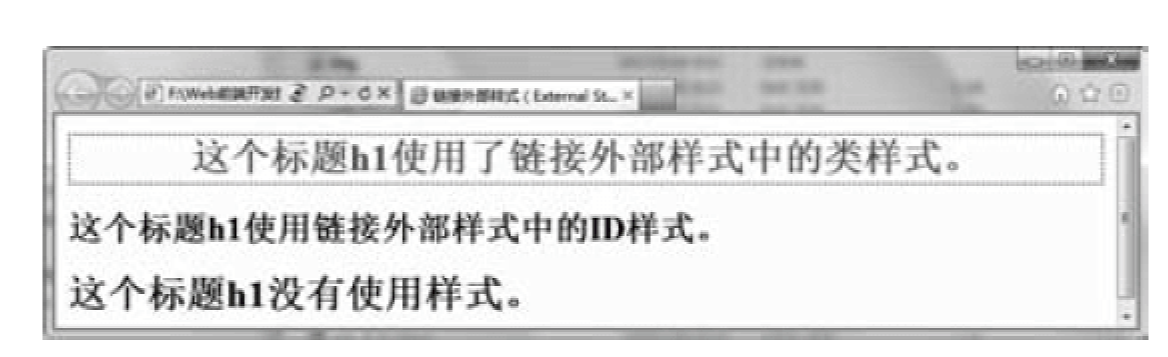



图 7-5　链接外部样式表

（3）代码解释。

代码中第 7 行在 head 标记中插入 link 标记链接外部样式表文件 out.css，属性 href 的值为 CSS 文件的路径，可以是绝对路径或相对路径。第 10 行引用了外部样式表中定义的类选择器 int_css，该 h1 标题字样式生效。第 11 行引用了外部样式表中定义的 ID 选择器 #h1_css，该 h1 标题字样式生效。

2）导入外部样式表

（1）基本语法。

```
<style type="text/css">
  @import  url("外部样式表文件1名称");
  @import  url("外部样式表文件2名称");
  选择器1{属性1：属性值1;属性2：属性值2;...}
  选择器2{属性1：属性值1;属性2：属性值2;...}
  ...
  选择器n{属性1：属性值1;属性2：属性值2;...}
</style>
```

（2）语法说明。

- 导入样式表必须在 style 标记内开头的位置定义，可以同时导入多个外部样式表，每条语句必须以“;”结束。一般导入外部样式写在最前面，内部样式写在后面。
- “@import”必须连续书写，即“@”和“import”之间不能留有任何空格。
- url（“外部样式表文件名称”）中的文件名称必须是全称，含后缀名.css，如 out.css。

【例 7-2-6】导入外部样式表的使用。代码如下所示，页面效果如图 7-6 所示。





视频讲解

图 7-6　导入外部样式表

```
1 <!-- edu_7_2_6.html -->
2 <!doctype html>
3 <html lang="en">
4     <head>
5         <meta charset="UTF-8">
6         <title>导入外部样式（External Style）</title>
7         <style type="text/css">
8             @import url("out.css");
9             @import url("out1.css");
10            @import url("out2.css");
11            #h2_css{
12                font-size:24px;      /*定义字体大小*/
13                font-style:italic;  /*定义字体样式*/
14        }
15        </style>
16    </head>
17    <body>
18        <h1 class="int_css">这个标题h1使用了导入外部样式表中的类样式(int_css)。
          </h1>
19        <h2 id="h2_css">这个标题h2使用内部样式中的ID样式(h2_css)。</h2>
20        <h2>这个标题h2没有使用样式，out1.css和out2.css未定义。</h2>
21    </body>
22 </html>
```

（3）代码解释。

代码中第 8 行～第 10 行通过“@import”导入三个外部样式表文件，分别是 out.css、

out1.css、out2.css。第 18 行引用导入外部样式表中的类选择器 int_css，第 19 行引用内部样式表中的 ID 样式 h2_css，第 20 行是默认样式。

外部样式表与内联样式表和内部样式表相比，具有以下优点：

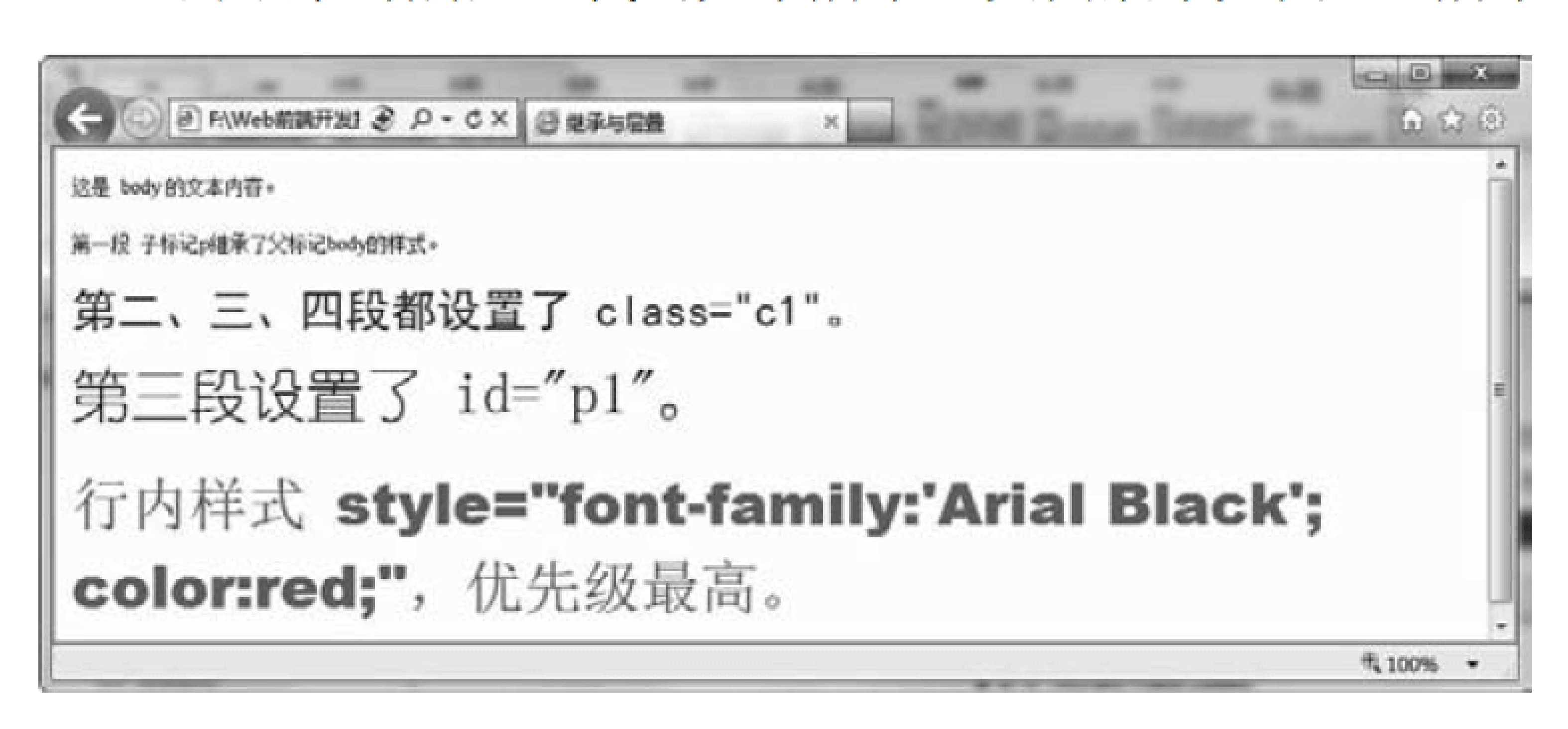
- 便于复用。一个外部 CSS 文件所定义的样式，可以被多个网页共用。
- 便于修改。修改样式只需要修改 CSS 文件，无须修改每个网页。
- 提高显示速度。样式写在网页里，网页文件变“胖”，增加网页传输的负担，降低网页显示速度。如果某 CSS 文件已被某网页引用并加载，则其他需要引用该 CSS 文件的网页时可以从缓存中直接读取该 CSS 文件，从而提高网页显示速度。

## 7.3　CSS 继承与层叠

CSS 继承即子标记会继承父标记的所有样式风格，并且可以在父标记样式风格的基础上再加以修改，产生新的样式，而子标记的样式风格完全不影响父标记。值得注意的是，并不是所有的属性都会自动传给子元素，有的属性不会继承父标记的属性值，例如边框属性就是非继承的。

CSS 的全称是“层叠样式表”，层叠特性和“继承”不一样，可以把层叠特性理解成“冲突”的解决方案，即对同一内容设置了多个不同类型样式产生冲突时的处理，CSS 规定如下优先级为：**行内样式 > id 样式 > class 样式 > 标记样式**。

**【例 7-3-1】**CSS 的继承与层叠。代码如下所示，页面效果如图 7-7 所示。






视频讲解

图 7-7　CSS 继承与层叠

```
<!-- edu_7_3_1.html -->
<!doctype html>
<html lang="en">
    <head>
        <meta charset="UTF-8">
        <title>继承与层叠</title>
        <style type="text/css">
            body{ font-size:12px;  }/*元素样式*/
            .c1{ font-size:28px; color:blue;font-family:"黑体";}/*class样式*/
            #p1,#p2{ font-family:"幼圆";font-size:36px;}/*id样式*/
        </style>
    </head>
    <body>
        这是 body 的文本内容。
```

```
15        <p>第一段 子标记p继承了父标记body的样式。</p>
16        <p class="c1">第二、三、四段都设置了 class="c1"。</p>
17        <p class="c1" id="p1">第三段设置了 id="p1"。</p>
18        <p class="c1" id="p2" style="font-family:'Arial Black';color:
          red;">行内样式 style="font-family:'Arial Black'; color:red;"，优先
          级最高。</p>
19    </body>
20 </html>
```

**代码解释**

代码中第 15 行定义段落 p 标记与 body 中的文本样式一致，说明它继承了第 8 行所设置的其父标记样式。第 16～18 行定义的三个段落 p 标记均设置了 class 属性，根据显示效果，说明 class 样式的优先级高于标记样式。第 17 行设置段落 p 标记的 id 和 class 属性。id 样式修改了字体为幼圆、字号为 36px，说明样式得到了应用，但颜色并没有发生变化，说明 id 样式优先级高于 class 样式。第 18 行定义的段落 p 标记，同样设置了 id 属性，同时增加了行内样式设置，效果显示其字号为 36px，说明 id 样式得到了应用，但字体变为 Arial Black，而不是幼圆，说明行内样式优先级高于 id 样式；字的颜色变为红色，说明行内样式优先级高于 class 样式，即行内样式的优先级最高。

## 7.4 综 合 实 例

设计 Web 页面时使用 CSS 来控制显示，将会使得页面结构清晰、代码量大大减少，而且便于维护，作为从事 Web 前端开发的工程师应该养成使用 CSS 实现页面内容和表现分离的编程习惯，并在实践中不断探索。

以“**Hoverbox 图像画廊**”（Hoverbox Image Gallery）为例，利用链接外部样式表 hoverbox.css 控制以无序列表方式排列的 5 行×4 列共 20 幅图像的样式，通过鼠标在某个图像上盘旋，实现大图像浏览。本例对原代码和样式文件进行了适当简化。

**【例 7-4-1】**Hoverbox 图像画廊程序。实现代码如下所示，页面效果如图 7-8 所示。




图 7-8　鼠标经过图片显示大图

- 主程序 edu_7_4_1.html。

```
<!-- edu_7_4_1.html -->
<!doctype html>
<html lang="en">
   <head>
      <meta charset="UTF-8">
      <title>Hoverbox Image Gallery</title>
      <link type="text/css" rel="stylesheet" href='hoverbox.css'  />
   </head>
   <body>
      <div id="" class="">
         <h1>鼠标经过图片显示大图(Hoverbox Image Gallery)</h1>
         <ul class="hoverbox">
            <li><a href="#">
             <img src="img/photo01.jpg" alt="description" class=
            "preview" />
               <img src="img/photo01.jpg" alt="description" /></a>
            </li>
            <li><a href="#">
             <img src="img/photo02.jpg" alt="description" class=
            "preview" />
               <img src="img/photo02.jpg" alt="description" /></a>
            </li>
            <li><a href="#">
             <img src="img/photo03.jpg" alt="description" class=
            "preview" />
               <img src="img/photo03.jpg" alt="description" /></a>
            </li>
            <li><a href="#">
             <img src="img/photo04.jpg" alt="description" class=
            "preview" />
               <img src="img/photo04.jpg" alt="description" /></a>
            </li>
            <li><a href="#">
              <img src="img/photo05.jpg" alt="description" class=
            "preview" />
               <img src="img/photo05.jpg" alt="description" /></a>
            </li>
            <li><a href="#">
             <img src="img/photo06.jpg" alt="description" class=
            "preview" />
               <img src="img/photo06.jpg" alt="description" /></a>
            </li>
            <li><a href="#">
             <img src="img/photo07.jpg" alt="description" class=
            "preview" />
               <img src="img/photo07.jpg" alt="description" /></a>
            </li>
            <li><a href="#">
             <img src="img/photo08.jpg" alt="description" class=
            "preview" />
               <img src="img/photo08.jpg" alt="description" /></a>
            </li>
            <li><a href="#">
             <img src="img/photo09.jpg" alt="description" class=
            "preview" />
               <img src="img/photo09.jpg" alt="description" /></a>
            </li>
            <li><a href="#">
             <img src="img/photo10.jpg" alt="description" class=
            "preview" />
```

```
                <img src="img/photo10.jpg" alt="description" /></a>
            </li>
        </ul>
    </div>
  </body>
</html>
```

- CSS 外部样式文件 hoverbox.css。

```
/* hoverbox.css */
*{                      /* 全局声明 */
   border: 0;
   margin: 0;
   padding: 0;
}
/* =Basic HTML, Non-essential
------------------------------------------------------------------*/
a{    text-decoration: none;}
div{                            /* 定义图层的样式*/
  width:720px;
  height:500px;
  margin:0 auto;
  padding:30px;
  text-align:center;            /* 定义内容居中显示  */
}
body{                           /* 定义主体样式 */
   position: relative;          /* 位置属性为相对的 */
   text-align:center;
}
h1{                             /* 定义h1的样式  */
   background: inherit;         /* 定义背景属性取值为继承  */
   border-bottom: 1px dashed #097;
   color: #000099;
   font: 17px Georgia, serif;
   margin: 0 0 10px;
   padding: 0 0 35px;
   text-align: center;
}
/* =Hoverbox Code
------------------------------------------------------------------*/
.hoverbox{cursor: default;list-style: none;}/* 去掉列表项前的符号 */
.hoverbox a{cursor: default;}
.hoverbox a .preview{display: none;}     /* 大图初始加载为不显示  */
.hoverbox a:hover .preview{              /* 派生选择器声明 */
   display: block;                       /* 以块方式显示 */
   position: absolute;                   /* 以绝对方式显示，图可以层叠 */
   top: -33px;                           /* 相对当前位置偏移量*/
   left: -45px;                          /* 相对当前位置偏移量*/
   z-index: 1;                           /* 表示在上层（原小图在底层）*/
}
.hoverbox img{                  /* 定义图像样式 */
   background: #fff;
   border-color: #aaa #ccc #ddd #bbb;
   border-style: solid;
   border-width: 1px;
   color: inherit;
```

```
48     padding: 2px;
49     vertical-align: top;
50     width: 100px;
51     height: 75px;
52 }
53 .hoverbox li{                    /* 定义列表项样式 */
54     background: #eee;            /* #eee等同于#eeeeee,以下格式相同*/
55     border-color: #ddd #bbb #aaa #ccc;
56     border-style: solid;
57     border-width: 1px;
58     color: inherit;
59     float: left;                 /* 设置图像向左浮动 */
60     display: inline;             /* 设置为行内显示 */
61     margin: 3px;
62     padding: 5px;
63     position: relative;          /* 位置为相对的方式 */
64 }
65 .hoverbox .preview{              /* 定义大图样式 */
66     border-color: #000;
67     width: 200px;
68     height: 150px;
69 }
70 ul{padding:40px;margin:0 auto; }/* 定义ul样式 */
```

**代码解释**

代码中第 10～54 行在 div 中插入一个无序列表，并在无序列表中插入 10 个列表项，在每一个列表项中插入两个图像超链接；其中一个图像在初始加载时通过.hoverbox a .preview 样式实现不显示，当鼠标在某一小图像上盘旋时通过在 hoverbox.css 中设置 position 和 z-index 属性的值来实现图像层叠，显示出大图像。本例仅显示 10 幅图像，分两行显示。

原始 Hoverbox 图像画廊页面 http://host.sonspring.com/hoverbox/。

# 本 章 小 结

本章介绍了 CSS 的基本概念以及如何使用 CSS 控制网页显示。

CSS 规则由选择器和声明组成，声明即“属性:属性值”对。选择器包括 id 选择器、类选择器、标记选择器、伪类选择器及属性选择器等，提供了不同的选取页面标记的方式。

根据 CSS 规则定义的位置不同，将 CSS 分为内联样式表、内部样式表、链接外部样式表以及导入外部样式表，其中内联样式表是在标记内设置 style 属性，且仅对该标记有效；内部样式表是在页面的 head 标记中加入 style 标记，在 style 标记里编写 CSS 规则，它对整个页面都有效；外部样式表是将 CSS 规则写在单独的文件里，要求该文件的后缀名为.css，称为 CSS 文件，需要应用规则的页面，通过 link 标记或者“@import”语句将独立的 CSS 文件引入到页面中，前者称为链接外部样式表，后者称为导入外部样式表。

CSS 继承性表明子标记将继承父标记的规则，CSS 层叠特性约定了规则冲突的解决方案。CSS 规定样式优先级从高到低为：行内样式 > id 样式 > class 样式 > 标记样式。

# 练习与实验

## 练习 7

**1. 选择题**

（1）CSS 的规则是由选择器和（　　）构成的。

（A）声明　　（B）属性　　（C）值　　（D）属性选择器

（2）下列选项中 CSS 规则书写正确的是（　　）。

（A）body:color=black　　（B）{body;color:black;}

（C）body{color:black;}　　（D）{body:color=black;}

（3）下列选项中正确定义所有 h3 标记内文字为特粗的是（　　）。

（A）<h3 style= " font-size:bolder; " >

（B）<h3 style= " font-weight:bolder; " >

（C）h3{font-size:bolder;}

（D）h3{font-weight:bolder;}

（4）在 CSS 中定义能多次引用样式的选择器是（　　）。

（A）超链接选择器　　（B）类选择器

（C）id 选择器　　（D）标记选择器

（5）下列选项中样式优先级最高的是（　　）。

（A）标记样式　　（B）id 样式　　（C）class 样式　　（D）行内样式

（6）下列选项中导入外部样式表正确的是（　　）。

（A）@import url("chu12015.css ")　　（B）@import " chu2015.css "

（C）<lik href=" chu12015.css "/>　　（D）@import url("chu12015.css ");

**2. 填空题**

（1）在 CSS 文件中，用#p1{}，在 HTML 中 p 标记内使用________属性引用样式；在 CSS 中使用.p2{}定义样式，在 HTML 中 p 标记内使用________属性来引用样式。

（2）引用外部 CSS 文件有两种方式：一是通过_______标记的_______属性；二是通过______标记内__________________来引用。

（3）CSS 文件的扩展名为____________。

**3. 简答题**

（1）简述属性选择器有几种定义方式。

（2）CSS 按照其定义位置可分为哪几种？分别如何使用？

（3）如何理解 CSS 的继承与冲突特性？

（4）简述 id 选择器与类选择器的异同点。

## 实验 7

1. 使用内联样式表及内部样式表，设计如图 7-9 所示的页面。设计要求如下：

（1）使用标题字和段落标记进行文字显示，在内部样式表中定义 body 标记内信息“居

中显示”、定义 p 标记字体为“隶书”。

（2）通过 p 标记的 style 属性定义字体大小属性（font-size）的值分别为 150%、200%、250%。“朝辞白帝彩云间，”不定义任何样式。

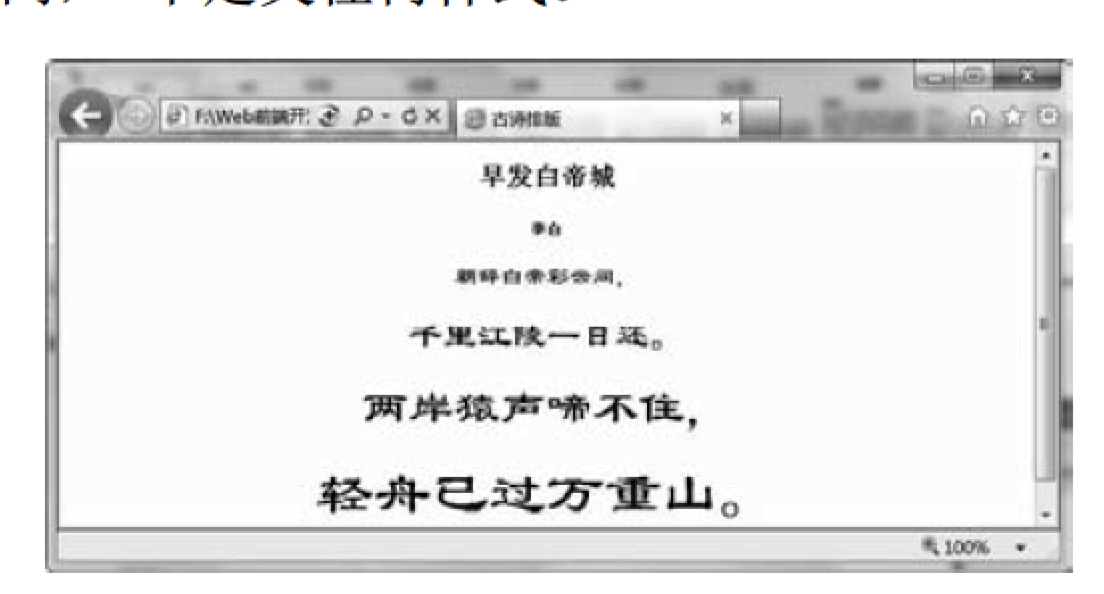



图 7-9　古诗排版效果图

2. 按如下要求设计“Web 前端开发工程师工作内容”页面，设计如图 7-10 所示的页面。要求如下：

（1）页面标题为：Web 前端开发工程师工作内容；

（2）页面题目：1 号标题字显示“Web 前端开发工程师工作内容”；3 号标题字显示“Web 前端工程师在不同的公司，会有不同的职能，但称呼都是类似的。”；

（3）采用无序列表显示工作内容，分四个方面，分别是“做网站设计、网页界面开发”“做网页界面开发”“做网页界面开发、前台数据绑定和前台逻辑的处理”“设计、开发、数据处理”；每一个列表项显示一种不同风格的工作内容，其中第一个列表项 ID（li1）样式为“斜体、加粗、24px、黑体”；第二个列表项类（li2）样式为“背景色#9999cc、字符间距 1px”；第三个列表项 ID（li3）样式为“字大小 18px、颜色红色”；第四个列表项行内样式“颜色#0000cc、背景色#c0c0c0、隶书”；

（4）定义全局样式为“楷体、蓝色”。




图 7-10　Web 前端开发工程师工作内容

# 第 8 章　DIV 与 SPAN

本章学习目标

**在网页设计过程中经常会需要对网页进行分区或切割成若干块，并在不同的分区或块中显示相关图、文、表等信息。除了表格、框架外，还有谁能胜任这项工作呢？只有图层 DIV 可以。DIV 就是为了简化页面布局，配合 CSS 完成精彩的页面布局设计。本章重点介绍 DIV 定义语法、属性语法、多 DIV 和 DIV 嵌套布局等方面的知识。**

**Web 前端开发工程师应掌握以下内容：**

- **掌握 DIV 标记的基本用法、常用属性。**
- **理解 DIV 嵌套与层叠的含义。**
- **掌握 SPAN 标记的语法，灵活使用 SPAN 标记。**
- **掌握 DIV 与 SPAN 标记在使用上的差异。**
- **学会使用 div+CSS 进行和简易页面布局。**

## 8.1　DIV 图层

图层是设计网页时用于定位元素或者布局的一种技术，它可以将图层里包含的内容放置到浏览器的任意位置，其包含的内容有文字、图像、动画甚至是图层。在一个网页文件中可以使用多个图层，图层可以嵌套、重叠，图层布局比表格的布局更加灵活。

### 8.1.1　DIV 定义

div（division/Section）是分区或分节的意思，这意味着它的内容自动地开始一个新行。图层 div 标记是一个块级标记，可定义文档中的分区或节。可以通过<div>的 class 或 id 应用额外的样式。div 标记是成对标记，以<div>开始，以</div>结束。

**1. 基本语法**

```
<div id="" class="" style="">块包含的内容</div>
```

**2. 语法说明**

div 标记的属性、取值及说明如表 8-1 所示。

**表 8-1　div 标记的属性、取值及说明**

| 属　　性 | 值 | 说　　明 |
|---|---|---|
| id | id | 规定元素的唯一 id |
| class | classname | 规定元素的类名（classname） |
| style | style_definition | 规定元素的行内样式（inline style） |

style 属性：设置图层的样式，未定义前通过浏览器查看不到效果。图层 style 属性的取值可以由多个“属性/属性值”对构成。其中主要属性有：

- position 属性——定义图层的定位方式，有 static、fixed、relative、absolute 四个属性值。常用 relative 和 absolute。若指定为 static 时，div 遵循 HTML 规则；若指定为 relative 时，可以用 top、left 来设置 div 在页面中的偏移，但是此时不可使用层叠；若指定为 absolute 时，可以用 top、left 对 div 进行绝对定位；若指定为 fixed 时，在 IE7 以上与 FireFox 中 div 的位置相对于屏幕固定不变，在 IE6 中没有效果。
- left、top 属性——定义图层左上角位置（左边距和上边距）。
- width、height 属性——定义图层的宽度和高度。
- float 属性——设置图层的浮动位置，可以向左、向右浮动或不浮动。
- clear 属性——清除图层内浮动，与浮动属性是一对作用相反的属性。可以清除向左、向右、左右两边浮动或允许浮动。
- z-index 属性——设置图层的层叠的上、下层关系，设置此属性以实现多个图层层叠的效果。z-index 值越大，图层的位置越高。子层始终位于父层之上。

div 标记的 style 属性的取值中属性、取值及说明表如表 8-2 所示。

**表 8-2 div 标记的 style 属性取值中属性、值及说明表**

| 属　性 | 值 | 说　明 |
|---|---|---|
| position | static | 表示静态定位，默认设置 |
| | absolute | 表示绝对定位，与位置属性配合使用 |
| | relative | 表示相对定位，图层不可层叠 |
| | fixed | 表示图层位置固定，不滚动 |
| border | 线粗细 线型 线颜色 | 边框，可以设置风格、粗细、颜色等属性 |
| background-color | rgb()\|十六进制数\|英文颜色名 | 背景颜色 |
| left | pixes\|% | 规定图层左边距离 |
| top | pixes\|% | 规定图层与顶部的距离 |
| width | pixes\|% | 规定图层的宽度 |
| height | pixes\|% | 规定图层的高度 |
| float | left\|right\|none | 允许浮动元素在左边、右边及不浮动 |
| clear | left\|right\|both\|none | 分别表示清除左边、右边、左右两边的浮动和允许左右两边有浮动 |
| z-index | auto\|数字 | 表示子图层会按照父层的属性显示\|无单位的整数或负数 |
| overflow | scroll\| visible \| auto \| hidden | 内容溢出控制。分别表示始终显示滚动条、不显示滚动条，但超出部分可见、内容超出时显示滚动条、超出时隐藏内容 |
| display | block\|inline\|none | 表示按块元素显示、行内方式显示和隐藏等 |

## 8.1.2 DIV 应用

div 标记通常设置 id 或 class 属性来引用定义的样式，把文档分割为独立的、不同的部分，对文档进行布局。

【例 8-1-1】div 标记的应用。代码如下所示，页面效果如图 8-1 所示。


视频讲解

```
<!-- edu_8_1_1.html -->
<!doctype html>
<html lang="en">
 <head>
  <meta charset="UTF-8">
      <style type="text/css">
          .inline_div{ display:inline;    }
          #div1{ background-color:green;
              width:300px;height:100px; float:left;    }
          #div3{ background-color:yellow; color:black;
              font-size:200%; clear:both;}
      </style>
    </head>
    <body>
      <div id="div1" class="inline_div">这是div1</div>
      <div class="inline_div">这是div2</div>
      <div id="div3">这是div3</div>
    </body>
</html>
```

上述代码中使用了三个 div 标记，从页面效果可以看出 div 是块标记，可以用来作文档分块；div1 与 div2 在一行显示，说明通过设置 display 属性可以改变其固有的性质。

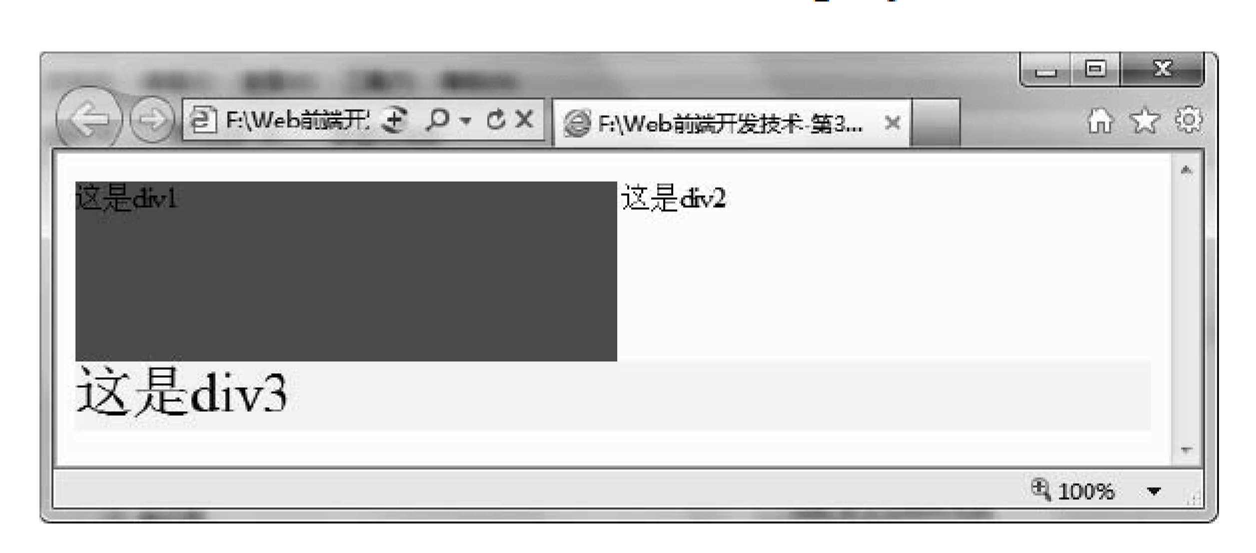



图 8-1　div 使用

## 8.2　图层嵌套与层叠

在图层中不仅可以包含文字、图像、动画等内容，还可以包含其他的图层，称为图层的嵌套。图层与图层之间可以不相交，也可以重叠，这就给页面布局带来了很大的灵活性，所以在设计网页时首先应设计好页面的结构，理清图层与图层之间的关系。

### 8.2.1　DIV 嵌套

多个 div 既可以单独使用，也可以互相包含，嵌套使用。一方面可以将页面分割成不同的块，块与块之间没有包含关系；另一方面又可以把功能相近的块组织到一个更大的块中，便于整体控制，即图层嵌套。

**【例 8-2-1】**div 的嵌套。代码如下所示，页面效果如图 8-2 所示。


视频讲解

```
<!-- edu_8_2_1.html -->
<!doctype html>
<html lang="en">
 <head>
  <meta charset="UTF-8">
```

```
        <style type="text/css">
            .inline_div{display:inline-block;/* 行内显示方式*/    }
            #wrap{ width:450px;height:250px; border:2px solid black;}
            #d1,#d2{height:100px; width:40%; background-color: green;
                    margin:20px;    }
            #d2{background-color:yellow; }
            #d3{height:100px; width:90%; border:2px solid black;
        background-color:#66ff33; margin:0 auto;    }
            h3{font-size:28px;color:#0033ff;}
        </style>
    </head>
    <body>
        <h3>图层嵌套的应用</h3>
        <div id="wrap">
            <div id="d1" class="inline_div">div1</div>
            <div id="d2" class="inline_div">div2</div>
            <div id="d3">div3</div>
        </div>
    </body>
</html>
```

上述代码中使用四个 div 标记演示其嵌套关系，外层<div id="wrap">里面包含三个<div>，其中<div id="d1">与<div id="d2">在同一行，<div id="d3">在第二行。




图 8-2　div 嵌套

## 8.2.2　DIV 层叠

多个 div 除了可以相互嵌套外，还可以层叠。div 层叠必须首先将 position 属性设置为 absolute，然后利用 z-index 属性控制层叠关系。

**【例 8-2-2】** div 的层叠。代码如下所示，页面效果如图 8-3 所示。



视频讲解

```
<!-- edu_8_2_2.html -->
<!doctype html>
<html lang="en">
 <head>
  <meta charset="UTF-8">
        <style type="text/css">
            body{margin:0;  /*margin表示边界*/    }
            div{position:absolute; /* 定位方式为绝对定位 */
                width:200px;height:200px;    }
            #d1{background-color:black;color:white;
```

```
            z-index:0;   /* 该图层在最下面 */        }
        #d2{background-color:red;top:25px; left:50px;
            z-index:1;   /* 该图层在中间 */        }
        #d3{ background-color:yellow; top:50px; left:100px;
            z-index:2;   /* 该图层在最上面 */}
      </style>
   </head>
   <body>
      <div id="d1" >div1</div>
      <div id="d2" >div2</div>
      <div id="d3" >div3</div>
   </body>
</html>
```




图 8-3　div 层叠

上述代码中使用了三个 div 标记，div 标记 position 属性值为 absolute（绝对定位），再设置 div 标记的宽度与高度；在子图层中定义 top、left 等属性的值对其进行偏移定位，多个 div 就可能重叠；通过 z-index 属性设置其层叠关系，运行效果说明 z-index 值最大的图层位于最上方。

## 8.3　div 标记与 span 标记

在使用 CSS 排版的页面中，div 标记和 span 标记是两个常用的标记。利用这两个标记，加上 CSS 对其样式的控制，可以很方便地实现各种效果。

**1．span 标记的使用**

div 标记是区块（block-level）容器标记，可以容纳段落、标题、表格、图像等各种 HTML 元素。只需对 div 标记进行样式控制，就可以对 div 内包含的各种元素进行样式控制。div 标记包含的元素会自动换行。

span 标记是行内标记，也是行内元素（inline element），同样可以包含 HTML 的各种元素，只不过其元素会在一行内显示。在它前后不会自动换行。span 标记没有结构上的意义，纯粹是应用样式，当其他行内元素都不适合时，就可以使用 span 元素。

1）基本语法

```
<span id="样式名称" class="样式名称">…</span>
```

2）语法说明

如果不给 span 标记应用样式，那么 span 标记包含的元素不会有任何视觉上的变化，

只有应用样式后，才会有效果。

**2．div 与 span 标记的区别**

div 和 span 标记默认情况下都没有对标记内的内容进行格式化或渲染，只有使用 CSS 来定义相应的样式时才会显示出不同。

（1）是否是块标记。div 标记是块标记，一般包含较大范围，在区域的前后会自动换行；而 span 标记是行内标记，一般包含范围较窄，通常在一行内，在区域外不会自动换行。

（2）是否可以互相包含。一般来说，div 标记可以包含 span 标记，但 span 标记不可以包含 div 标记。

但是块标记和行标记不是绝对的，通过定义 CSS 的 display 属性可以相互转化，display 属性的取值如表 8-3 所示。

**表 8-3　display 属性的取值及说明表**

| 属　性　值 | 说　　明 |
|---|---|
| none | 此元素不会被显示 |
| inline | 将对象设置为行内元素，在行内显示 |
| block | 将对象设置为块级元素，以块状显示，自动换行 |
| inline-block | 将对象设置为行内块标记 |
| inherit | 规定应该从父元素继承 display 属性的值 |

**【例 8-3-1】**块元素和行元素的相互转化，代码如下所示，页面效果如图 8-4 所示。

视频讲解

```
<!-- edu_8_3_1.html -->
<!doctype html>
<html lang="en">
 <head>
  <meta charset="UTF-8">
       <style type="text/css">
           div{background-color:#f6f6f6;color:#000000;
                height:2em;margin:2px;  /*margin表示边界*/}
           .inline_disp{display:inline;     /*改变div显示方式*/}
           .block_disp{display:block;  /*改变span显示方式*/
              height:4em;background-color:rgb(200,200,200);
              margin:2px;  /*margin表示边界*/     }
       </style>
    </head>
    <body>
       <div id="d1">这是div1</div>
       <div id="d2">这是div2</div>
       <span id="s1">这是span1</span>
       <span id="s2">这是span2</span>
       <div id="d3" class="inline_disp">这是div3</div>
       <div id="d4" class="inline_disp">这是div4</div>
       <span id="s3" class="block_disp">这是span3，在使用CSS排版的页面中，
       div标记和span标记是两个常用的标记。利用这两个标记，加上CSS对其样式的控制，
       可以很方便地实现各种效果。</span>
       <span id="s4" class="block_disp">这是span4，在使用CSS排版的页面中，
       div标记和span标记是两个常用的标记。利用这两个标记，加上CSS对其样式的控制，
       可以很方便地实现各种效果。</span>
    </body>
</html>
```




图 8-4　div 与 span 标记显示方式转换

上述代码中第 16～19 行说明了 div 和 span 固有的特征，即 div 是块标记，span 是行内标记。第 20 行和第 21 行说明设置 display 属性为 inline，可以将块标记 div 设置成行内显示；第 22 行和第 23 行说明设置 display 属性为 block，可以将行内标记 span 改变成块形式显示。

## 8.4　综 合 实 例

本例以“苏州百特电器有限公司”首页作为参照网站，如图 8-5 所示，使用 DIV+CSS 完成页面布局设计，设计效果与原网站（http://www.better-vac.com/）相似（省略图像幻灯片播放部分）。布局图如图 8-6 所示，页面效果如图 8-7 所示。




图 8-5　苏州百特电器有限公司网站首页

### 1. 页面布局规划

根据图 8-5 页面布局效果，我们很容易看出这是标准 4 行 3 列布局样式。使用布局绘图软件画出布局图，如图 8-6 左图所示。

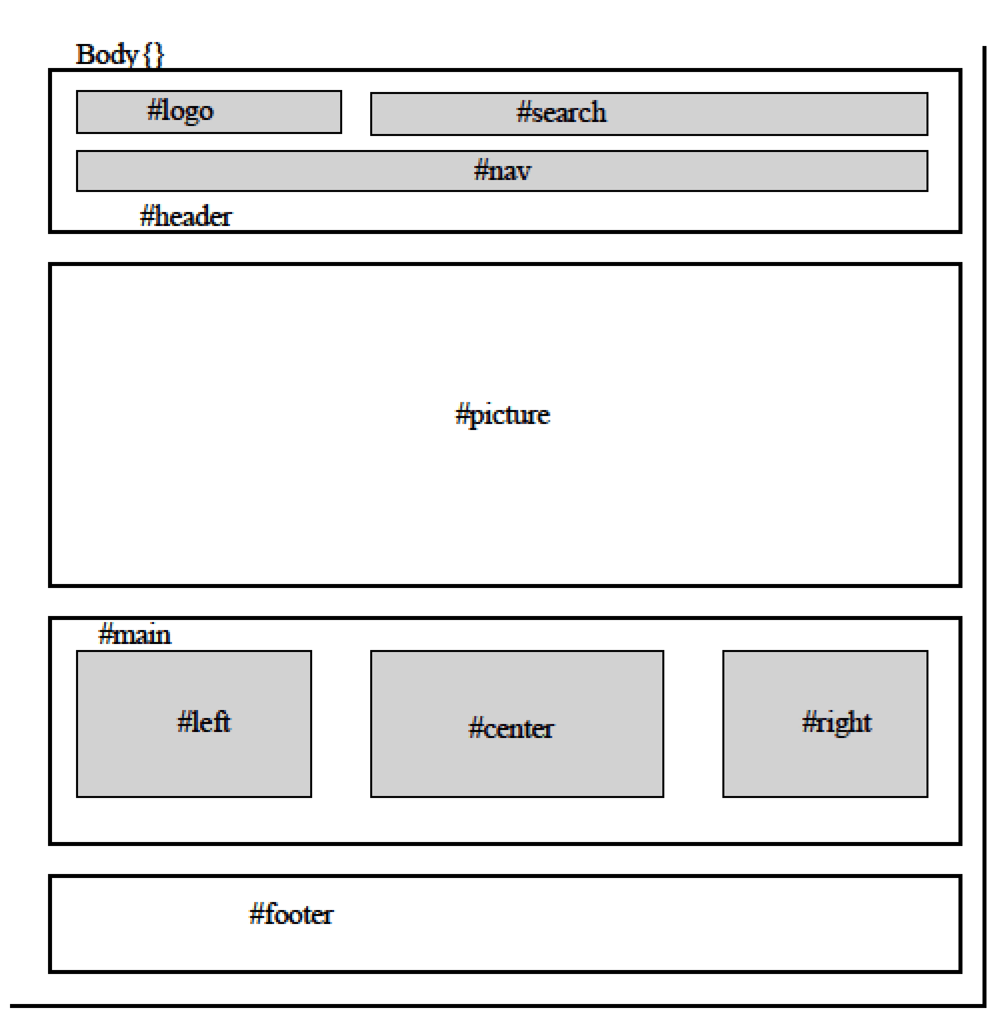





图 8-6　网站设计 div 布局

## 2. 写出 div 结构代码

新建一个 HTML 文档，编写如下的 div 嵌套结构代码。

- header 部分的 div 结构。

```
<div id="header" class="">
    <div id="logo" class=""></div>
    <div id="search" class=""></div>
    <div id="nav" class=""></div>
</div>
```

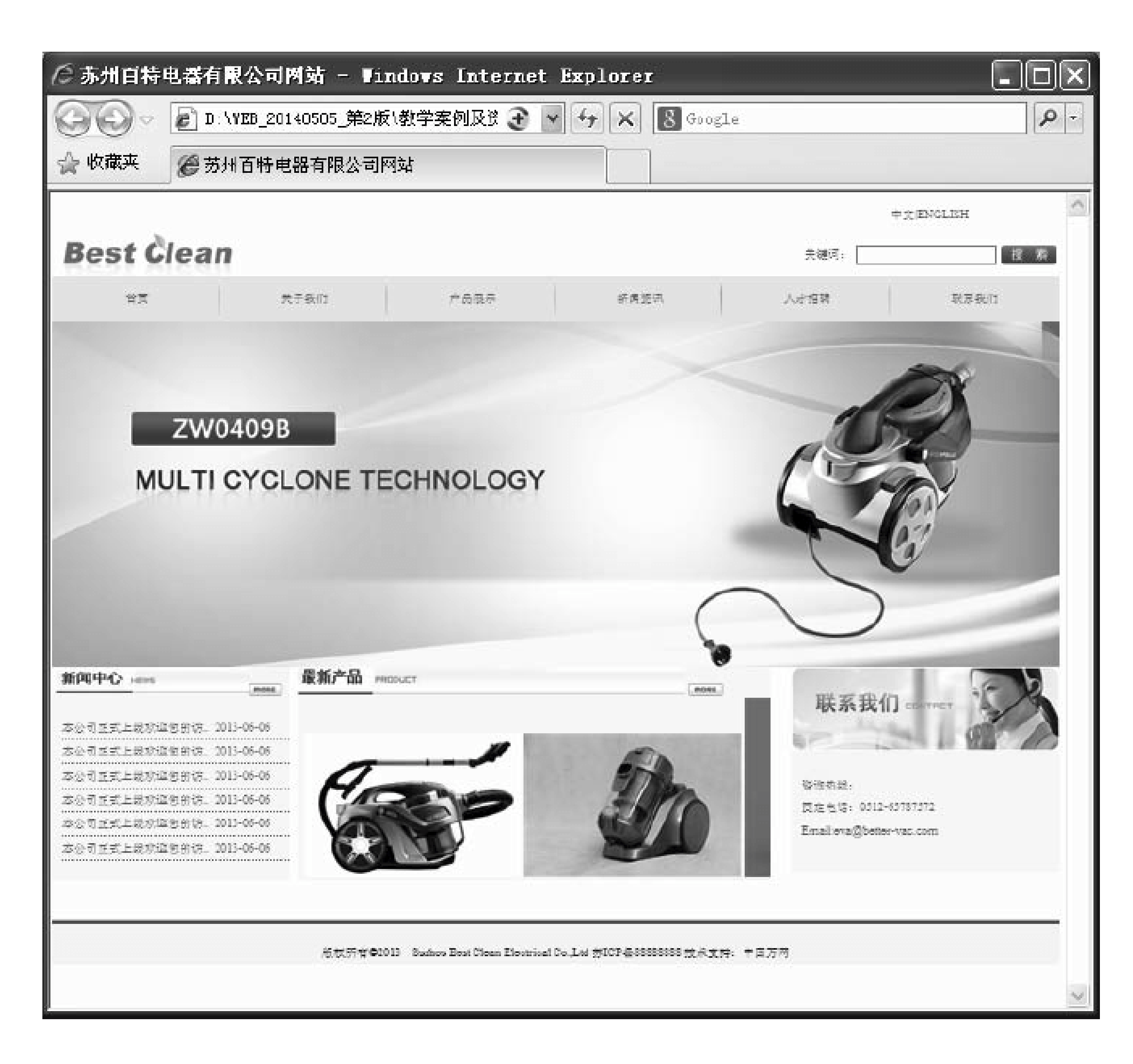



图 8-7　网站仿真设计效果

- picture 部分的 div 结构。

```
<div id="picture" class=""></div>
```

- mian 部分的 div 结构。

```
<div id="main" class="">
    <div id="left" class=""></div>
    <div id="center" class=""></div>
    <div id="right" class=""></div>
</div>
```

- footer 部分的 div 结构。

```
<div id="footer" class=""></div>
```

### 3. 编写 best.css 文件

编写 CSS 文件，定义相应的 ID 样式，形成页面初步布局，效果如图 8-6 右图所示。

```
/*  best.css  */
*{font-size:12px;font-family:Times New Roman;color:#828282;}
body{width:984px;height:800px;
   padding:2px;margin:0 auto;}
#header{width:984px; height:120px;background:#FFFFFF;}
#logo{width:199px; height:80px; float:left;
 background:#FFFFFF url("logo.png") no-repeat left bottom;}
#search{width:785px; height:80px;
 background:#FFFFFF; float:right;}
#nav{clear:both; width:984px; height:40px; background:#EBEBEB;}
#picture{width:984px; height:337px; background:#828282;}
#main{width:984px; height:250px; border-bottom:5px ridge #DEDEDE; }
#left{width:240px;height:200px;background:#EEFFDD;
        float:left; border-right:10px solid #FFFFFF;}
#center{width:466px;height:200px;float:left;
        background:#EEFFDD; border-right:5px solid #FFFFFF;}
#right{width:263px;  height:200px;background:#EEFFDD; float:right; }
#footer{clear:both; width:984px; height:40px;  background:#F7F7F7;
           text-align:center; }
table{text-align:center;line-height:1.5em;}
a:link,a:visited,a:active{text-decoration:none;}
a:link,a:visited,a:active{color:#828282;}
#nav a:hover{color:#Bf0000;text-decoration:none;}
#nav td{width:165px;height:40px;
    text-align:center;vertical-align:middle;}
#line{background:url("line.png") no-repeat right center;}
ul{list-style:none;margin-left:10px;line-height:1.8em;}
#left li{border-bottom:1px dotted #009900;}
#left a:hover{color:#Bf0000;text-decoration:underline;}
input{vertical-align:middle;padding:2px auto;}
#footer p{margin-top:20px;text-align:center;color:#333333;}
```

### 4. 编写 HTML 代码

```
<!-- edu_8_4_1.html -->
<!doctype html>
<html lang="en">
   <head>
      <meta charset="UTF-8">
      <link rel="stylesheet" href="best.css" type="text/css">
      <title>苏州百特电器有限公司网站</title>
   </head>
   <body>
```

```
        <div id="header" class="">
            <div id="logo" class=""></div>
            <div id="search" class="">
            <table height="80px" align="right">
                <tr>
                    <td colspan="3">中文|ENGLISH</td>
                </tr>
                <tr>
                    <td>关键词：</td>
                    <td><input type="text" name="" size="25"></td>
                    <td><input type="image" src="serach.png"></td>
                </tr>
            </table>
        </div>
        <div id="nav" class="">
            <table width="100%" height="40px" align="center">
                <tr>
                    <td id="line"><a href="">首页</a></td>
                    <td id="line"><a href="">关于我们</a></td>
                    <td id="line"><a href="">产品展示</a></td>
                    <td id="line"><a href="">新闻资讯</a></td>
                    <td id="line"><a href="">人才招聘</a></td>
                    <td><a href="">联系我们</a></td>
                </tr>
            </table>
        </div>
        </div>
        <div id="picture" class=""><img src="big_zw9021.png"></div>
        <div id="main" class="">
            <div id="left" class="">
                <img src="xwzx.jpg" border="0" alt=""><input type="image" src="more.png">
                <ul>
                    <li><a href="">本公司正式上线欢迎您的访..</a>  2013-06-06</li>
                    <li><a href="">本公司正式上线欢迎您的访..</a>  2013-06-06</li>
                    <li><a href="">本公司正式上线欢迎您的访..</a>  2013-06-06</li>
                    <li><a href="">本公司正式上线欢迎您的访..</a>  2013-06-06</li>
                    <li><a href="">本公司正式上线欢迎您的访..</a>  2013-06-06</li>
                    <li><a href="">本公司正式上线欢迎您的访..</a>  2013-06-06</li>
                </ul>
            </div>
            <div id="center" class="">
                <img src="zxcp.jpg" border="0" alt="">
                <input type="image" src="more.png">
                <marquee onmouseover="this.stop()" onmouseout="this.start()">
                    <img src="ZW0409B.jpg" border="0" alt="ZW0409B">
                    <img src="ZW9021.jpg" border="0" alt="ZW9021">
                    <img src="ZL9012.jpg" border="0" alt="ZL9012">
                    <img src="ZW9020.jpg" border="0" alt="ZW9020">
                </marquee>
            </div>
```

```
            <div id="right" class="">
                <img src="lianxi.png" border="0" alt="">
                <ul>
                    <li>咨询热线：</li>
                    <li>固定电话：0512-65787572</li>
                    <li>Email:eva@better-vac.com</li>
                </ul>
            </div>
        </div>
        <div id="footer" class="">
            <p>版权所有&copy;2013    Suzhou Best Clean Electrical Co.,Ltd   苏ICP备88888888  技术支持：中国万网
            </p>
        </div>
    </body>
</html>
```

# 本 章 小 结

本章介绍了 div 及 span 标记的基本语法以及两个标记在使用时的区别。一般而言，div 标记是块级标记，span 标记是行内标记；div 标记可以自动换行，而 span 标记则不可以；div 标记可能包含 div 和 span 标记，但 span 标记不可以包含 div 标记。但这两个标记的外在表现可以通过设置 display 属性的值为 inline 或 block 来实现转换。

div、span 标记必须配合 CSS 使用才能实现精确定位页面上每一个元素。通过 id、class 来引用已经定义的 CSS 文件中类选择器、ID 选择器及其他选择器。

# 练习与实验

## 练习 8

**1. 选择题**

（1）下列选项中为行内标记的是（　　）。

（A）<p></p>　（B）<div></div>　（C）<span></span>　（D）<pre></pre>

（2）下列选项中能够实现两个图层 div 同时向右浮动的是（　　）。

（A）div{float:right; }　（B）div{float:none;}

（C）div{float:left; }　（D）div{clear:both;}

（3）下列能够将 div 标记由块显示方式改为行内显示方式的选项是（　　）。

（A）div{overflow:hidden;}　（B）div{display:inline;}

（C）div{display:block;}　（D）div{display:none;}

（4）多个图层要实现层叠的必要条件是 position 属性的值必须是（　　）。

（A）static　（B）relative　（C）absolute　（D）fixed

（5）下列选项中能够清除 div 左右两边浮动的属性是（　　）。

（A）clear　（B）display　（C）overflow　（D）float

**2. 填空题**

（1）在 HTML 文件中，定义图层的标记是________；定义<span></span>标记样式可以通过定义三个属性来实现，它们分别是________、________、__________。

（2）定位一个图层的位置可以通过四个属性来定位，为 left、________、width、________。

（3）设置图层层叠关系可以通过设置__________属性来实现，其属性值越大，图层越层叠在上层。但前提条件是需要将_________属性的值设置为 absolute。

**3. 简答题**

（1）简述<div>标记与<span>标记的异同点。

（2）如何设置多个图层层叠关系？

## 实验 8

1. 利用<div>及<span>标记设计如图 8-8 所示的页面，写出实现的 HTML 代码。要求使用链接外部样式表。设计要求如下：

（1）编写外部样式表文件，名称为 exp_8_1.css，采用链接外部样式表的方法。

（2）加载图像文件名为 exp_8_1.jpg。

（3）定义两个图层，最外层图层包含一个图像和一个子图层，在子图层内采用无序列表显示四行文字。

（4）对“央视”“腾讯”“跨界融合 开放共赢”三个词采用 span 标记定义加粗样式。

（5）对“联建杯”定义斜体、加粗、大小 24px。样式如下：

```
.it{font-style:italic;font-size:24px;font-weight:bold;}
```

2. 按如下要求设计“匾牌设计”页面，如图 8-9 所示。要求如下：页面标题为“匾牌设计”；页面内容为一个图层中嵌入一个段落，段落的内容“海纳百川 有容乃大”；段落的样式为“斜体、特粗、70px 大小、行高 1.5 倍、隶书”；图层 div 的 #div0 样式：“宽度 800px、高度 100px、边框宽度 20px、线型 outset、颜色 #ff0000、填充 20px、边距 100px”；页面所有内容居中显示（body 标记的样式）。




图 8-8 新闻效果图




图 8-9 匾牌设计

# 第 9 章　CSS 样式属性

本章学习目标

CSS 最大的作用是实现网页的内容与表现的分离，要让 CSS 发挥这一用途必须掌握 CSS 控制页面的文字、图像、颜色、列表等样式的属性是什么，然后再对这些元素的属性进行设置，使之达到精确控制页面每一元素的目的。本章重点介绍 CSS 盒子模型结构及构成盒子模型的边界（Margin）、边框（Border）、填充（Padding）、内容（Content）等相关属性（简称 MBPC），进而达到灵活运用 CSS+DIV 进行页面布局的目标。

Web 前端开发工程师应掌握以下内容：

- **熟悉 CSS 样式设置中常用的单位。**
- **掌握控制文字、文本、背景、色彩、列表等样式的属性及设置方法。**
- **理解 CSS 盒子模型。**
- **掌握边框、边界、填充及内容等属性及设置方法。**

## 9.1　CSS 属性值中的单位

设置 CSS 属性值的难点在于单位的选用。它覆盖范围较广，从长度单位到颜色单位，再到 URL 地址等。单位的取舍在很大程度上取决于用户的显示器和浏览器，不恰当地使用单位会给页面布局带来很多麻烦，因此属性值的单位设置需要慎重考虑，合理使用。

### 9.1.1　绝对单位

绝对单位在网页中很少使用，一般多用在传统平面印刷中，但在特殊场合使用绝对单位是很有必要的。绝对单位包括英寸、厘米、毫米、磅和 pica（皮卡）。

- 英寸（in）：使用最广泛的长度单位（1in=2.54cm）。
- 厘米（cm）：生活中最常用的长度单位。
- 毫米（mm）：在研究领域使用比较广泛。
- 磅（pt）：在印刷领域使用较为广泛，也称为点。CSS 也常用 pt 设置字体大小，12 磅的字体等于 1/6 in 大小（1pt=1/72in）。
- pica（pc）：在印刷领域使用较多，1pc=12pt，所以也称为 12 点活字。

### 9.1.2　相对单位

相对单位与绝对单位相比显示大小不是固定的，它所设置的对象受屏幕分辨率、视觉

区域、浏览器设置以及相关元素的大小等因素影响。CSS 属性值中经常使用的相对单位包括 em、ex、px、%。

**1. em**

em 表示元素的字体高度，它能够根据字体的 font-size 属性值来确定单位的大小，例如：

```
p{font-size:24px;line-height:2em;  /*行高为48px*/}
```

代码中设置字体大小为 24px，行高为 2em，即是字体大小的 2 倍，所以行高为 48px。如果 font-size 的单位为 em，则 em 的值将根据父元素的 font-size 属性值来确定。

**2. ex**

ex 表示以所使用的字体中小写字母 x 的高度作为参考。在实际使用中，浏览器将通过 em 的值除以 2 得到 ex 的值。

**3. px**

px 表示根据屏幕像素点来确定的。这样不同的显示分辨率就会使相同取值的像素单位所显示出来的效果截然不同。在实际设计过程中，建议 Web 前端开发工程师多使用相对单位 em，且在某一类型的单位上使用统一的单位。如在网站中可以统一使用 px 或 em。

**4. 百分比%**

百分比也是一个相对单位值。百分比的值总是通过另一个值来进行计算，一般参考父元素中相同属性的值。例如，如果父元素宽度为 200px，子元素的宽度为 50%，则子元素实际宽度为 100px。举例如下：

```
p{font-size:250%;line-height:150%;}
```

## 9.2 CSS 字体样式

使用 font 标记对页面元素进行字体、字号大小、颜色的设置所产生的样式也是有限，不够丰富。而在 CSS 中，通过 font 属性可以设置丰富多彩的文字样式。该属性是复合属性，所包含的子属性如表 9-1 所示。

**表 9-1 font 子属性表**

| 属　性 | 说　明 |
|---|---|
| font-size | 设置字体的大小 |
| font-style | 设置字体的风格 |
| font-variant | 设置小型的大写字母字体 |
| font-family | 设置字体名 |
| font-weight | 设置字体的粗细 |

### 9.2.1 字体大小 font-size 属性

font-size 属性用于设置文本字体的大小，其值可以是绝对或相对值。绝对值将文本设置为指定的大小，不允许用户在所有浏览器中改变文本大小，这不利于可用性，但在确定输出的物理尺寸时很有用；相对值是相对于周围的元素来设置大小，允许用户在浏览器中改变文本大小。

1．基本语法

```
font-size:绝对大小|相对大小|关键字;
```

2．语法说明

（1）绝对大小：可以使用 in、cm、mm、pt、pc 等单位为 font-size 属性赋值。

（2）相对大小：可以使用 em、ex、px、%等单位为 font-size 属性赋值。

网页通常是为了浏览而不是印刷，建议用相对单位来定义字号，例如 px，W3C 推荐使用 em 尺寸单位，从而可以在所有浏览器中调整文本字体大小。

font-size 属性值也可以通过关键字来指定大小，font-size 属性值关键字有 xx-small、x-small、small、medium、large、x-large、xx-large 等。在不同的终端设备上浏览的效果会有些差异。

## 9.2.2 字体样式 font-style 属性

在 HTML 中，使用<em></em>、<i></i>标记可将文字设置成为斜体。在 CSS 中可以使用 font-style 属性设置字体的风格，例如显示斜体字样。

1. 基本语法

```
font-style: normal | italic | oblique
```

2. 语法说明

font-style 属性取值及说明如表 9-2 所示。

表 9-2 font-style 属性取值及说明表

| 属 性 值 | 说 明 |
| --- | --- |
| normal | 表示不使用斜体，是 font-style 属性的默认值 |
| italic | 表示使用斜体显示文字 |
| oblique | 表示使用倾斜字体显示 |

## 9.2.3 字体系列 font-family 属性

在 CSS 中使用 font 属性可以设置丰富的字体，美化页面的外观。其中 font-family 专门用于设置字体名称系列。

1. 基本语法

```
font-family:字体1,字体2,…,字体n
```

2. 语法说明

属性值为多个字体名称时，可以使用逗号（,）分隔。浏览器依次查找字体，只要存在就使用该字体，不存在将会继续找下去，以此类推，直到最后一种字体，仍不存在则使用默认字体（宋体）。如果字体名称中出现空格，必须使用双引号将字体括起来，例如 Times New Roman。

【例 9-2-1】设置字体大小、样式及字体名称。代码如下所示，页面效果如图 9-1 所示。

视频讲解

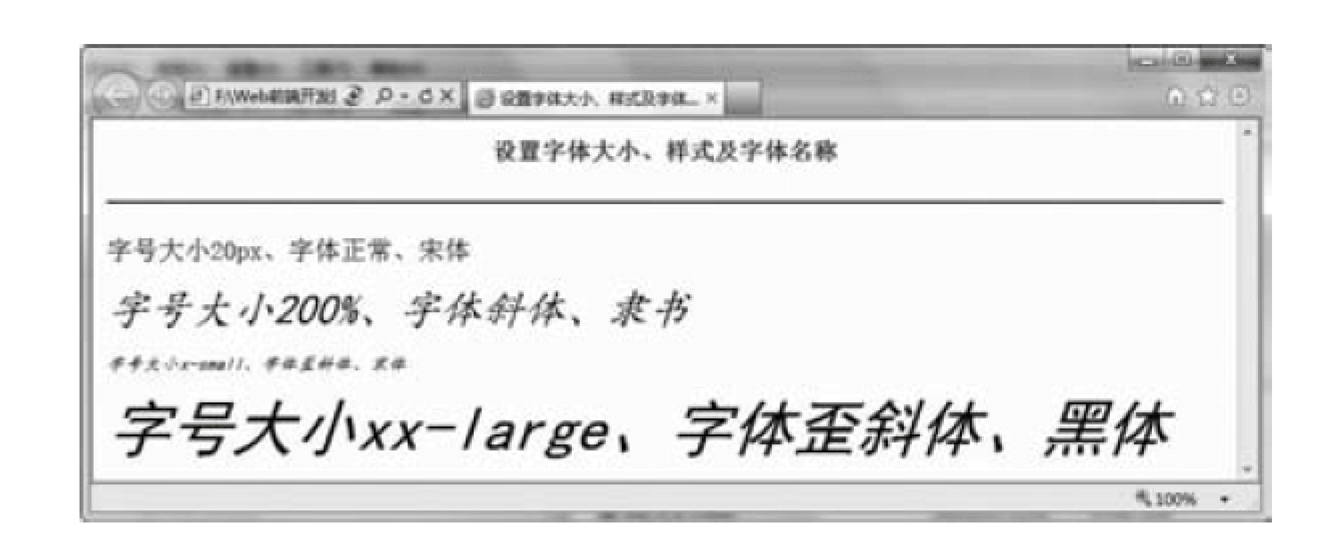



图 9-1　设置字体大小、样式及字体名称

```
<!--edu_9_2_1.html -->
<!doctype html>
<html lang="en">
    <head>
        <meta charset="UTF-8">
        <title> 设置字体大小、样式及字体名称 </title>
        <style type="text/css">
            h3{text-align:center;color:#3300ff;}
            hr{color:#660066;}
            #p1{font-size:20px;font-style:normal;font-family:宋体;}
            #p2{font-size:200%;font-style:italic;font-family:楷体,隶书;}
            #p3{font-size:x-small;font-style:oblique;font-family:楷体,宋
            楷体;}
            #p4{font-size:xx-large;font-style:oblique;font-family:黑体,
            隶书,楷体_gb2312;}
        </style>
    </head>
    <body>
        <h3>设置字体大小、样式及字体名称</h3>
        <hr>
        <p id="p1">字号大小20px、字体正常、宋体</p>
        <p id="p2">字号大小200%、字体斜体、隶书</p>
        <p id="p3">字号大小x-small、字体歪斜体、宋体</p>
        <p id="p4">字号大小xx-large、字体歪斜体、黑体</p>
    </body>
</html>
```

### 9.2.4　字体变体 font-variant 属性

font-variant 属性用于设置字体变体，主要用于设置英文字体，实际上是设置文本字体是否为小型的大写字母。

**1. 基本语法**

```
font-variant: normal | small-caps
```

**2. 语法说明**

font-variant 属性的参考值如表 9-3 所示。

表 9-3　font-variant 属性取值表

| 属　性　值 | 说　　明 |
|---|---|
| normal | 表示正常的字体，是 font-variant 属性的默认值 |
| small-caps | 表示使用小型的大写字母字体 |

### 9.2.5 字体粗细 font-weight 属性

在 HTML 中使用<strong></strong>或<b></b>标记来设置字体加粗。在 CSS 中可以使用 font-weight 属性用于设置文本字体的粗细。

**1. 基本语法**

```
font-weight: normal | bold | bolder | lighter |100|200|…|900
```

**2. 语法说明**

font-weight 属性的参考值如表 9-4 所示。

表 9-4 font-weight 属性取值表

| 属 性 值 | 说 明 |
|---|---|
| normal | 表示正常的字体，是 font-weight 属性的默认值 |
| bold | 表示标准的粗体 |
| bolder | 表示特粗体（为相对参数） |
| lighter | 表示细体（为相对参数） |
| 整数 | 取值为 100、200、……、900 来表示粗细程度，100 表示最细、400 等价于 normal、700 等价于 bold |

### 9.2.6 字体 font 属性

font 属性是复合属性，一次完成多个字体属性的设置，包括字体粗细、风格、字体变体、大小/行高及字体名称。

**1. 基本语法**

```
font:font-style font-weight font-variant font-size/line-height font-family
```

**2. 语法说明**

利用 font 属性一次完成多个字体属性的设置时，属性值与属性值之间必须使用空格隔开。前三个属性值可以不分先后顺序，默认为 normal。大小和字体名称系列必须显式指定，先设置大小，再设置字体系列。需要设置行高时，可以写在字体大小的后面，中间用“/”分隔，行高为可选的属性。font 属性可以继承。

【例 9-2-2】设置字体变体、粗细、复合属性。代码如下所示，页面效果如图 9-2 所示。

视频讲解

```
<!--edu_9_2_2.html -->
<!doctype html>
<html lang="en">
   <head>
       <meta charset="UTF-8">
       <title> 设置字体变体、粗细、复合属性 </title>
       <style type="text/css">
           h3{text-align:center;color:#3300ff;}
           hr{color:#660066;}
           #p1{font-variant:normal;font-weight:lighter;}
           #p2{font-variant:small-caps;font-weight:bold;}
           #p3{font-weight:600;font:italic 28px/40px 幼圆;}
           #p4{font:italic  bolder small-caps 24px/1.5em 黑体;}
       </style>
```

```
15     </head>
16     <body>
17         <h3>设置字体变体、粗细、复合属性 </h3>
18         <hr>
19         <p>此段文字正常显示Welcome to you!</p>
20         <p id="p1">此段文字Welcome to you!正常、较细字体。 </p>
21         <p id="p2">设置小型大写字母、字体标准粗体。</p>
22         <p  id="p3">设置字体粗细度为600、斜体、大小28px、行高50px、字体幼圆</p>
23         <p  id="p4">设置字体风格斜体、特粗、小型大写字母HTML、字号24px/行高1.5em、
          字体黑体</p>
24     </body>
25 </html>
```

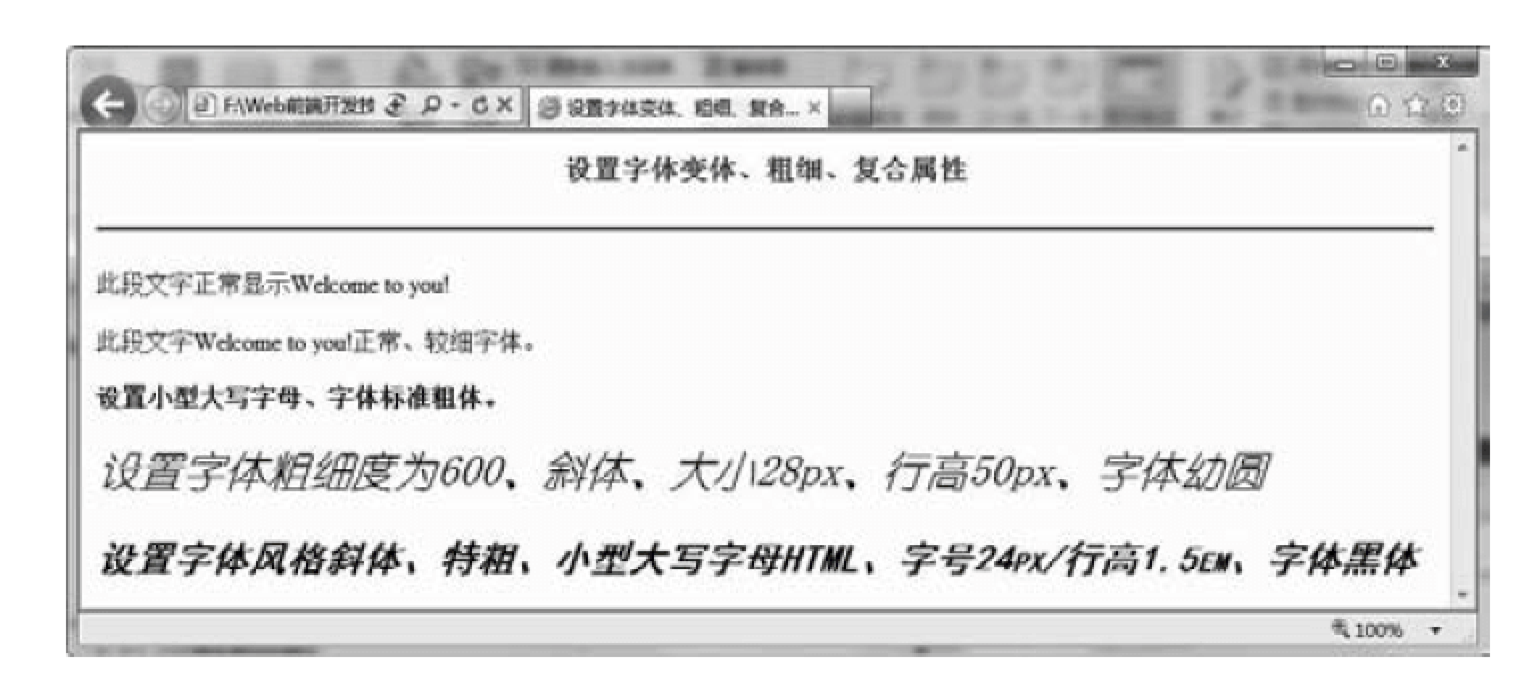



图 9-2　设置字体变体、粗细、复合属性

上述代码中第 7～14 行在 head 标记中插入内部样式表，并定义标题字 h3、水平分隔线 hr 标记样式和 4 个段落 id 样式。第 10 行定义字体正常、较细样式；第 11 行定义字体小型大写字母、标准粗体；第 12 行、第 13 行分别采用 font 复合属性定义了段落的样式。程序运行后，第 20～23 行分别应用样式 p1、p2、p3、p4，效果如页面中文字所示。

# 9.3　CSS 文本样式

在 CSS 中，不仅可以设置文字字体、大小、粗细、风格等，还可以对文本显示进行更精细排版设置。

## 9.3.1　字符间距 letter-spacing 属性

letter-spacing 间距属性可以设置字符与字符之间的距离。

**1. 基本语法**

```
letter-spacing:normal|长度单位
```

**2. 语法说明**

normal 表示默认间距，长度一般为正数，也可以使用负数，取决于浏览器是否支持。

word-spacing 属性主要针对英文单词，letter-spacing 属性对中文、英文字符串均起作用。

## 9.3.2　行距 line-height 属性

line-height 用于设置行与行之间的距离。

**1．基本语法**

```
line-height : normal | length
```

**2．语法说明**

normal：默认行高。

length：百分比、数字。由浮点数字和单位标识符组成的长度值，允许为负值。其百分比取值是基于字体的高度尺寸。

## 9.3.3 首行缩进 text-indent 属性

在 HTML 中段落的首行往往需要通过插入四个“ ”才能实现首行空两个字符的排版格式，而在 CSS 中可以使用 text-indent 属性来设置首行缩进量。

**1. 基本语法**

```
text-indent: 长度单位|百分比单位
```

**2. 语法说明**

长度单位可以使用绝对单位和相对单位，也可以使用百分比单位。

【例 9-3-1】设置字符间距、行距及首行缩进。代码如下所示，页面效果如图 9-3 所示。

视频讲解

```
<!--edu_9_3_1.html -->
<!doctype html>
<html lang="en">
    <head>
        <meta charset="UTF-8">
        <title> 设置字符间距、行高及首行缩进</title>
        <style type="text/css">
            h3{text-align:center;color:#3300ff;}
            hr{color:#660066;}
            #p1{letter-spacing:2px;line-height:1em;text-indent:2em;}
            #p2{letter-spacing:4px;line-height:1.5em;text-indent:3em;}
            #p3{letter-spacing:6px;line-height:2em;text-indent:4em;
            word-spacing:10px;}
        </style>
    </head>
    <body>
        <h3>设置字符间距、行高及首行缩进</h3>
        <hr>
        <p id="p1">[字符间距2px、行高1em、首行缩进2em]昨天上午，南京国际博览中
        心金陵会议中心内欢声笑语，春意盎然，省委、省政府在这里举行春节团拜会。省领
        导罗志军、李学勇、张连珍等与各界人士1000多人欢聚一堂，共迎传统新春佳节，向
        全省人民致以节日问候和美好祝福。</p>
        <p id="p2">[字符间距4px、行高1.5em、首行缩进3em]昨天上午，南京国际博览
        中心金陵会议中心内欢声笑语，春意盎然，省委、省政府在这里举行春节团拜会。省
        领导罗志军、李学勇、张连珍等与各界人士1000多人欢聚一堂，共迎传统新春佳节，
        向全省人民致以节日问候和美好祝福。</p>
        <p id="p3">[字符间距6px、行高2em、首行缩进4em、单词间距10px]昨天上午，
        南京国际博览中心金陵会议中心内欢声笑语，春意盎然，省委、省政府在这里举行春
        节团拜会。Chinese leader Xi Jinping has urged the Communist Party
        of China (CPC) to be more tolerant of criticism and receptive to
        the views of non-communists.</p>
    </body>
</html>
```

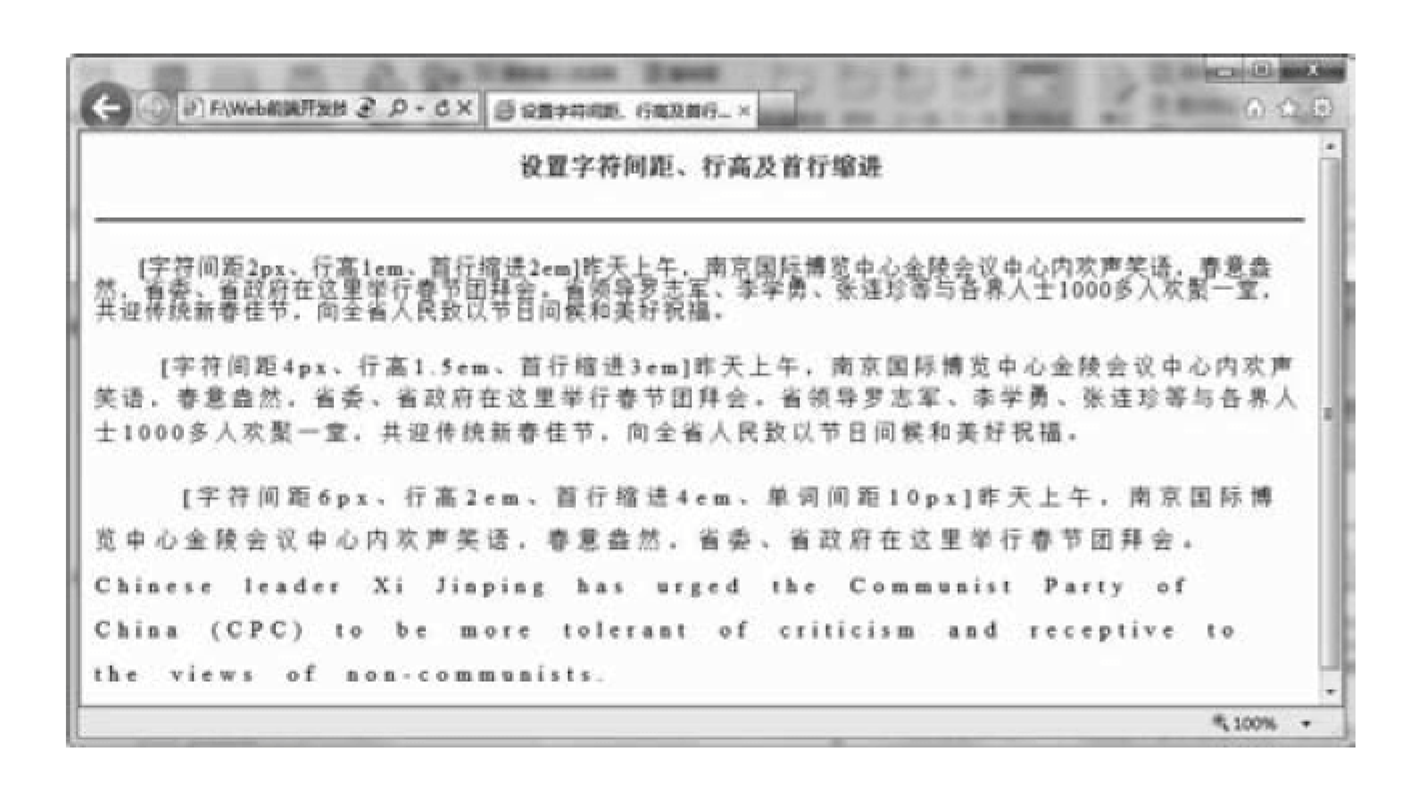



图 9-3 设置字符间距、行距及首行缩进

上述代码中第 18 行设置字符间距 2px、行高 1em、首行缩进 2em；第 19 行设置字符间距 4px、行高 1.5em、首行缩进 3em；第 20 行设置字符间距 6px、行高 2em、首行缩进 4em、单词间距 10px。页面效果截然不同。

### 9.3.4 字符装饰 text-decoration 属性

字符装饰 text-decoration 属性主要用来完成文字加上画线、下画线、删除线等效果。

**1. 基本语法**

```
text-decoration : none| underline | overline | line-through
```

**2. 语法说明**

none：表示文字无装饰。underline：表示文字加下画线。line-through：表示文字加删除线。overline：表示文字加上画线。

### 9.3.5 英文大小写转换 text-transform 属性

利用 text-transform 属性以转换英文大小写。

**1. 基本语法**

```
text-transform: capitalize| uppercase | lowercase| none
```

**2. 语法说明**

capitalize：将每个单词的第一个字母转换成大写，其余不转换。

uppercase：转换成大写；lowercase：转换成小写。

none：不转换。

**【例 9-3-2】**设置文字装饰及大小写转换。代码如下所示，页面效果如图 9-4 所示。

```
<!-- edu_9_3_2.html -->
<!doctype html>
<html lang="en">
   <head>
       <meta charset="UTF-8">
       <title> 设置文字装饰及大小写转换</title>
       <style type="text/css">
           h3{text-align:center;color:#3300ff;}
           hr{color:#660066;}
```



视频讲解

```
10            #p1{text-decoration:underline;text-transform:capitalize;}
11            #p2{text-decoration:line-through;text-transform:lowercase;}
12            #p3{text-decoration:overline;text-transform:uppercase;}
13        </style>
14    </head>
15    <body>
16        <h3>设置文字装饰及大小写转换</h3>
17        <hr>
18        <p id="p1">[文字下画线、首字母大写capitalize]Chinese leader Xi
          Jinping has urged the Communist Party of China (CPC) to be more
          tolerant of criticism and receptive to the views of non-
          communists.</p>
19        <p id="p2">[文字删除线、字母小写lowercase]Chinese leader Xi Jinping
          has urged the Communist Party of China (CPC) to be more
          tolerant of criticism and receptive to the views of non-
          communists.</p>
20        <p id="p3">[文字上画线、字母大写uppercase]Chinese leader Xi Jinping
          has urged the Communist Party of China (CPC) to be more
          tolerant of criticism and receptive to the views of non-
          communists.</p>
21    </body>
22 </html>
```

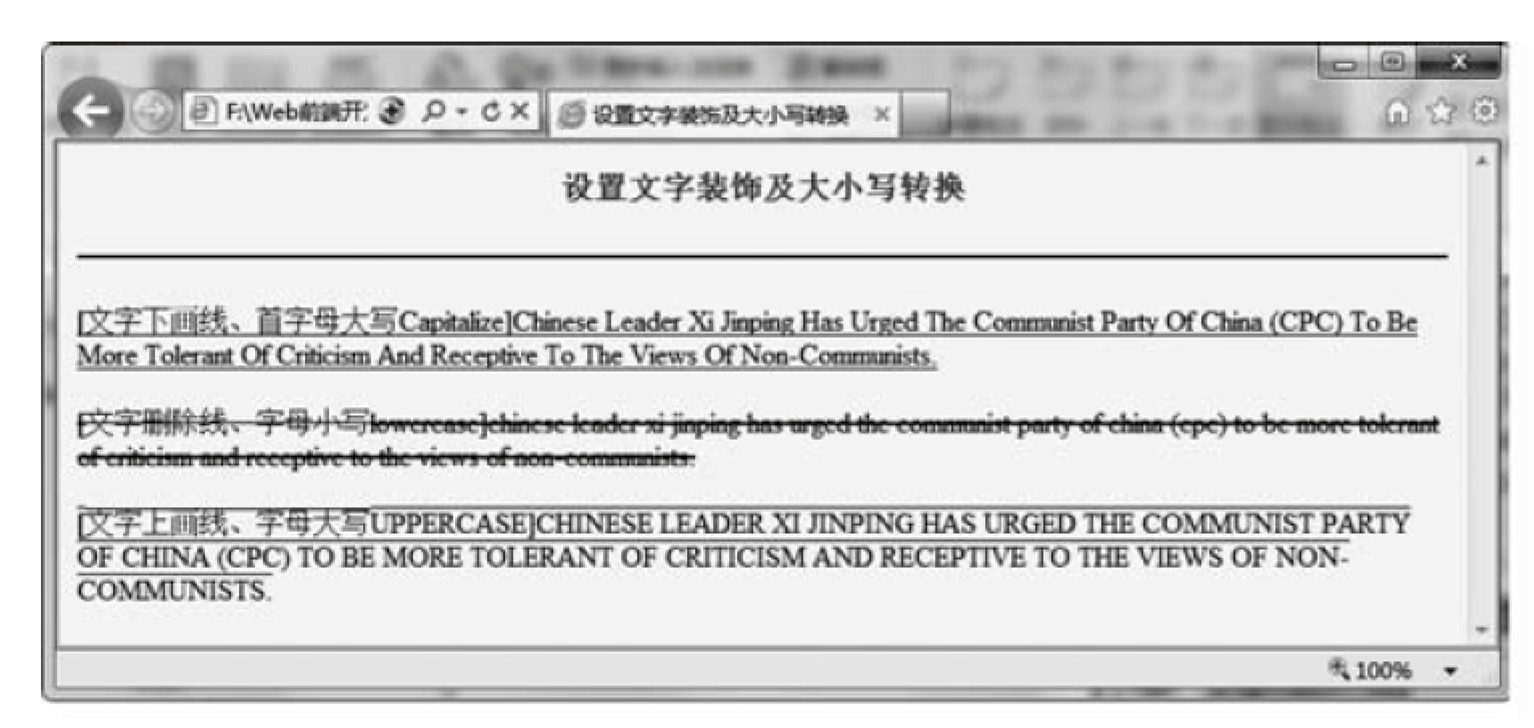



图 9-4　设置文字装饰及大小写转换

上述代码中第 18 行设置文字下画线、首字母大写 capitalize；第 19 行设置文字删除线、字母小写 lowercase；第 20 行设置了文字上画线、字母大写 uppercase。页面效果截然不同。

## 9.3.6　水平对齐 text-align 属性

text-align 属性规定元素的水平对齐方式。

**1. 基本语法**

```
text-align: left | right | center | justify
```

**2. 语法说明**

left：表示左对齐，默认值；right：表示右对齐；center：表示居中；justify：表示两端对齐。

## 9.3.7　垂直对齐 vertical-align 属性

vertical-align 属性以设置元素的垂直对齐方式。

### 1. 基本语法

```
vertical-align: top |middle |bottom|text-top|text-bottom
```

### 2. 语法说明

语法中常用属性值及说明如表 9-5 所示，还有一些不常用的属性值未列入其中。

表 9-5　vertical-align 常用属性值表

| 属　性　值 | 说　　明 |
| --- | --- |
| top | 把元素的顶端与行中最高元素的顶端对齐 |
| middle | 把此元素放置在父元素的中部 |
| bottom | 把元素的顶端与行中最低元素的顶端对齐 |
| text-top | 把元素的顶端与父元素字体的顶端对齐 |
| text-bottom | 把元素的底端与父元素字体的底端对齐 |

【例 9-3-3】设置内容对齐方式。代码如下所示，页面效果如图 9-5 所示。



视频讲解

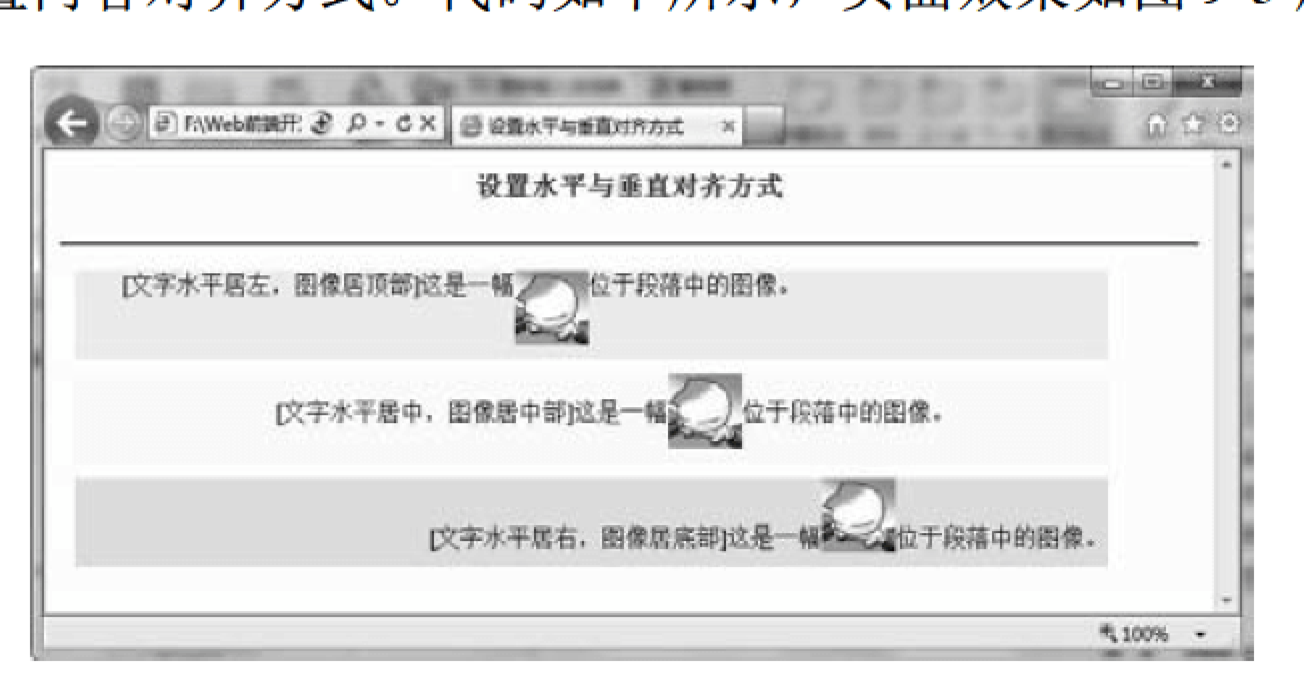



图 9-5　设置水平与垂直对齐方式

```
<!-- edu_9_3_3.html -->
<!doctype html>
<html lang="en">
    <head>
        <meta charset="UTF-8">
        <title> 设置水平与垂直对齐方式</title>
        <style type="text/css">
            h3{text-align:center;color:#3300ff;}
            hr{color:#660066;}
            #div1{margin:10px;width:700px;height:60px;background:#ccffcc;
            text-indent:2em;text-align:left;}
            #div2{margin:10px;width:700px;height:60px;background:#ffffcc;
          text-indent:2em;text-align:center;}
            #div3{margin:10px;width:700px;height:60px;background:#99ff99;
            text-indent:2em;text-align:right;}
            img{width:50px;height:50px;}
            #img1{vertical-align:text-top;}
            #img2{vertical-align:middle;}
            #img3{vertical-align:text-bottom;}
        </style>
    </head>
    <body>
        <h3>设置水平与垂直对齐方式</h3>
        <hr>
        <div id="div1" class="">
```

```
23          <p>[文字水平居左，图像居顶部]这是一幅<img id="img1" src="eg_cute.
            gif">位于段落中的图像。</p>
24        </div>
25        <div id="div2" class="">
26          <p>[文字水平居中，图像居中部]这是一幅<img id="img2" src="eg_cute.
            gif">位于段落中的图像。</p>
27        </div>
28        <div id="div3" class="">
29          <p>[文字水平居右，图像居底部]这是一幅<img id="img3" src="eg_cute.
            gif">位于段落中的图像。</p>
30        </div>
31    </body>
32 </html>
```

上述代码中第 22～24 行 div1 内设置文字水平居左，图像居顶部；第 25～27 行 div2 内设置文字水平居中，图像居中部；第 28～30 行 div3 内设置文字水平居右，图像居底部。

# 9.4　CSS 颜色与背景

网页设计中结构和内容仅是一方面，没有色彩的页面再精致也很难吸引人。CSS 中对于色彩、图像的设置也比较丰富，功能也很强大。

## 9.4.1　颜色 color 属性

color 属性用于设置元素字体的色彩，该属性的语法比较简单，但有多种取值，分别是颜色英文名称、rgb()函数、十六进制数等形式。

**1. 基本语法**

```
color : rgb（r%, g%, b%）|rgb（r, g, b）|#FFFFFF | #3FE |colorname
```

**2. 语法说明**

（1）颜色名称。使用 red、blue、yellow 等 CSS 预定义的表示颜色的参数。CSS 预定义 17 种颜色，常用的预定义颜色如表 9-6 所示。

（2）rgb()函数。使用 rgb（r, g, b）或 rgb（r%, g%, b%），字母 R 或 r、G 或 g、B 或 b 分别表示颜色分量红色、绿色、蓝色，前者参数的取值为 0～255，后者参数的取值为 0～100。

（3）十六进制数。使用“#rrggbb”或“#rgb”的形式，每位十六进制数的取值范围为 0～F，如#FFC0CB 表示 pink，#3DF 效果与#33DDFF 相同。

**表 9-6　颜色名称、函数及数值**

| 颜色名称 | 十六进制数 | rgb 百分数 | rgb 整数 |
|---|---|---|---|
| Black | #000000 | rgb (0%,0%,0%) | rgb (0,0,0) |
| White | #FFFFFF | rgb (100%,100%,100%) | rgb (255,255,255) |
| Red | #FF0000 | rgb (100%,0%,0%) | rgb (255,0,0) |
| Yellow | #FFFF00 | rgb (100%,100%,0%) | rgb (255,255,0) |
| Lime | #00FF00 | rgb (0%,100%,0%) | rgb (0,255,0) |
| Aqua | #00FFFF | rgb (0%,100%,100%) | rgb (0,255,255) |
| Blue | #0000FF | rgb (0%,0%,100%) | rgb (0,0,255) |

续表

| 颜色名称 | 十六进制数 | rgb 百分数 | rgb 整数 |
|---|---|---|---|
| Fuchsia | #FF00FF | rgb (100%,0%,100%) | rgb (255,0,255) |
| Gray | #808080 | rgb (50%,50%,50%) | rgb (128,128,128) |
| Silver | #c0c0c0 | rgb (75%,75%,75%) | rgb (192,192,192) |
| Maroon | #800000 | rgb (50%,0%,0%) | rgb (128,0,0) |
| Olive | #808000 | rgb (50%,50%,0%) | rgb (128,128,0) |
| Green | #008000 | rgb (0%,50%,0%) | rgb (0,128,0) |
| Teal | #008080 | rgb (0%,50%,50%) | rgb (0,128,128) |

## 9.4.2　背景 background 属性

background 属性用于设置指定元素（标记）的背景色彩、背景图案等，其子属性如表 9-7 所示。

**表 9-7　background 子属性表**

| 属　　性 | 说　　明 |
|---|---|
| background-color | 用于对指定元素设置背景颜色 |
| background-image | 用于对指定元素设置背景图案 |
| background-repeat | 在背景图案小于指定元素的情况下，是否重复填充图案 |
| background-attachment | 用于指定设置的背景图案在元素滚动时是否一起滚动 |
| background-position | 用于指定背景图案的起始位置 |

**1. 背景颜色 background-color 属性**

在 HTML 中，可以使用标记的 bgcolor 属性来设置背景色。在 CSS 中则使用 background-color 来设置网页的背景颜色。语法同 color 类似。

**2. 背景图像 background-image 属性**

background-image 属性用于设置指定元素的背景图案。

1）基本语法

```
background-image : url("图像文件名称")|none
```

2）语法说明

none：表示不用图像作为背景。url（"图像文件名称"）：表示图像的相对或绝对路径，如果图像文件和 CSS 文件在同一目录下，则可以直接使用图像文件名称。

**【例 9-4-1】**设置页面文字颜色及背景图像。代码如下所示，页面效果如图 9-6 所示。

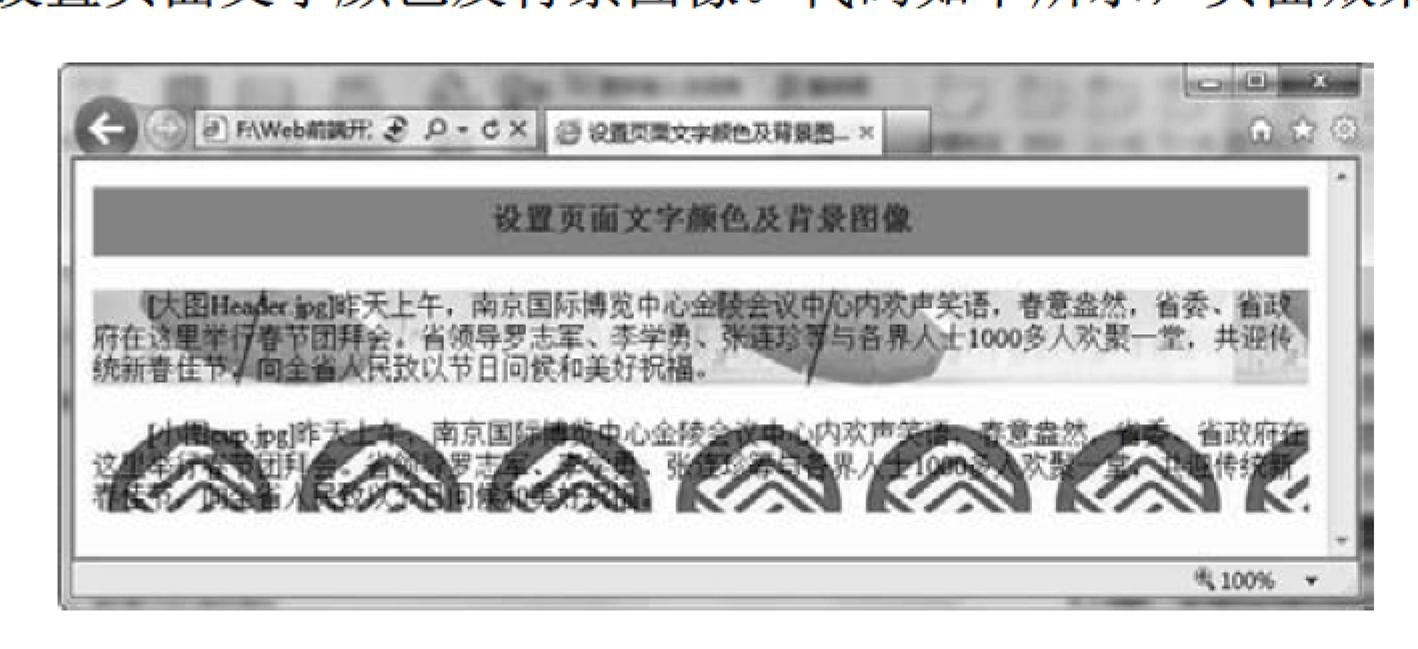



图 9-6　设置背景图像及颜色

视频讲解

```
1 <!-- edu_9_4_1.html -->
2 <!doctype html>
3 <html lang="en">
4     <head>
5         <meta charset="UTF-8">
6         <title>设置页面文字颜色及背景图像</title>
7         <style type="text/css">
8             h3{color:#0000ff;background-color:#9999ff;
              text-align:center;padding:10px;}
9             #p1{text-indent:2em;background-image:url("Header.jpg");}
10            #p2{text-indent:2em;background-image:url("cup.jpg");}
11        </style>
12    </head>
13    <body>
14        <h3>设置页面文字颜色及背景图像</h3>
15        <p id="p1">[大图Header.jpg]昨天上午，南京国际博览中心金陵会议中心内欢声
          笑语，春意盎然，省委、省政府在这里举行春节团拜会。省领导罗志军、李学勇、张连
          珍等与各界人士1000多人欢聚一堂，共迎传统新春佳节，向全省人民致以节日问候和
          美好祝福。
16        </p>
17        <p id="p2">[小图cup.jpg]昨天上午，南京国际博览中心金陵会议中心内欢声笑
          语，春意盎然，省委、省政府在这里举行春节团拜会。省领导罗志军、李学勇、张连珍
          等与各界人士1000多人欢聚一堂，共迎传统新春佳节，向全省人民致以节日问候和美
          好祝福。
18        </p>
19    </body>
20 </html>
```

上述代码中第 15 行应用 id 样式 p1，设置背景图像为 Header.jpg；第 17 行应用 id 样式 p2，设置背景图像为 cup.jpg，由于图像本身比较小，所以背景图像在水平方向重复填充了。

**3．背景图像重复 background-repeat 属性**

background-repeat 属性用于设置背景图案的重叠覆盖方式。

1）基本语法

```
background-repeat: repeat | no-repeat | repeat-x | repeat-y
```

2）语法说明

- repeat：使用背景图像完全填充元素大小的空间。
- repeat-x：使用背景图像在水平方向从左到右填充元素大小的空间。
- repeat-y：使用背景图像在垂直方向从上到下填充元素大小的空间。
- no-repeat：不使用背景图像重复填充元素。

**4．背景附件 background-attachment 属性**

background-attachment 背景附件属性设置背景图像是否随着滚动条一起滚动。

1）基本语法

```
background-attachment : scroll | fixed
```

2）语法说明

- scroll：表示在文字页面滚动时，背景附件一起滚动。

- fixed：表示在文字页面滚动时，背景附件固定不滚动。

**5．背景图像位置 background-position 属性**

background-position 属性用于设置背景图像的具体的起始位置。

1）基本语法

```
background-position: 参数1参数2
```

2）语法说明

图像的位置一般需要设置两个参数，且用空格分隔。两个参数的单位可以是百分比、长度单位或关键字。第一个参数表示水平位置，第二个参数表示垂直位置。也可以只设置一个参数，另一个参数自动为 50%或居中位置。参数取值如表 9-8 所示。

**表 9-8　background-position 属性值及说明**

| 属　性　值 | 说　　明 |
|---|---|
| left\|center\|right | 表示水平方向居左、居中、居右三个不同的位置 |
| top\|center\|bottom | 表示垂直方向顶部、中部、底部三个不同的位置。如果仅规定了一个值，另一个值将是 center |
| x%　y% | x%表示水平位置，y%表示垂直位置。左上角是 0%　0%，如果仅规定了一个值，另一个值将是 50% |
| xpos　ypos | xpos 表示水平位置，ypos 表示垂直位置。左上角是 0　0，如果仅规定了一个值，另一个值将是 50% |

**6．background 复合属性**

背景 background 是复合属性，可以使用它一次性完成背景颜色、图像、重复、位置和附件的设置。

1）基本语法

```
background:background-color background-image background-repeat
background-position  background-attachment
```

2）语法说明

语法中属性值的设置参考各属性进行设置。

【例 9-4-2】设置背景图像重复、位置与附件的应用。代码如下所示，页面效果如图 9-7 所示。




图 9-7　设置背景图像重复、位置与附件

视频讲解

```
1 <!-- edu_9_4_2.html -->
2 <!doctype html>
3 <html lang="en">
4    <head>
5       <meta charset="UTF-8">
6       <title>设置背景图像、位置与附件</title>
7       <style type="text/css">
8          h3{color:#ffffff;background-color:#6600ff;
         text-align:center;padding:10px;}
9          #p1{
10             background-image:url("Header.jpg");
11             background-repeat: no-repeat;
12             background-position:center center;}
13         #p2{
14             background-image:url("cup.jpg");
15             background-attachment:fixed;}
16         #p3{width:100%;height:150px;
17             background:#99ccff url("cup.jpg") no-repeat center center; }
18      </style>
19   </head>
20   <body>
21      <h3>设置背景图像、位置与附件</h3>
22      <p id="p1">[图像水平垂直居中]昨天上午，南京国际博览中心金陵会议中心内欢声
      笑语，春意盎然，省委、省政府在这里举行春节团拜会。省领导罗志军、李学勇、张连
      珍等与各界人士1000多人欢聚一堂，共迎传统新春佳节，向全省人民致以节日问候和
      美好祝福。</p>
23      <p id="p2">[图像水平居左到顶、固定]昨天上午，南京国际博览中心金陵会议中心
      内欢声笑语，春意盎然，省委、省政府在这里举行春节团拜会。省领导罗志军、李学勇、
      张连珍等与各界人士1000多人欢聚一堂，共迎传统新春佳节，向全省人民致以节日问
      候和美好祝福。</p>
24      <p id="p3">[背景复合属性应用]昨天上午，南京国际博览中心金陵会议中心内欢声
      笑语，春意盎然，省委、省政府在这里举行春节团拜会。省领导罗志军、李学勇、张连
      珍等与各界人士1000多人欢聚一堂，共迎传统新春佳节，向全省人民致以节日问候和
      美好祝福。</p>
25   </body>
26 </html>
```

上述代码中定义了三个 id 样式，p1 定义背景图像不重复且水平和垂直均居中，p2 定义背景图像附件不随滚动条移动，p3 定义宽度和高度，并采用复合属性 background 设置背景颜色、图像、重复、位置等。第 22 行应用 id 样式 p1，网页中图像不重复且水平、垂直均居中显示；第 23 行应用 id 样式 p2，网页中第 2 个段落背景图像重复填充整个区域，且在浏览器窗口缩小的情况下背景图像不随滚动条移动；第 24 行应用 id 样式 p3，网页中第 3 个段落设置宽度为 100%、高度为 150px、背景颜色、图像、不重复、位置居中等。

## 9.5　CSS 列表样式

HTML 中常用列表有三种类型，分别是无序列表、有序列表和定义列表。在实际应用中，常使用无序列表来实现导航和新闻列表的设计；使用有序列表实现条文款项的表示；使用定义列表来制作图文混排的排版模式。列表对于设计有语义的 XHTML 文档非常重要。CSS 中提供了 list-style-type、list-style-image、list-style-position、list-style 属性来改变列表符号的样式。

## 1. 基本语法

```
list-style-type: 属性值;  /* 设置列表类型，共9种 */
list-style-image: url("图像文件名称")|none;  /* 设置列表替代图像 */
list-style-position:outside|inside;  /* 设置图像位置 */
list-style: list-style-type list-style-image list-style-position; /* 复合属性 */
list-style:none url("smallico1.bmp") outside ;/* 复合属性的一个应用 */
```

## 2. 语法说明

list-style-type 属性取值如表 9-9 所示。

**表 9-9　list-style-type 属性说明表**

| 属　性　值 | 说　　明 |
|---|---|
| disc | 实心圆● |
| circle | 空心圆○ |
| square | 实心方块■ |
| decimal | 阿拉伯数字 123… |
| lower-roman | 小写罗马数字 I ii iii … |
| upper-roman | 大写罗马数字 I II III IV … |
| lower-alpha | 小写英文字母 abc… |
| upper-alpha | 大写英文字母 ABC… |
| none | 不使用项目符号 |

list-style-image 属性通过 url（"图像文件名称"）来加载图像，如果图像与 CSS 文件在同一目录，则直接使用图像文件名。属性值为 none 表示不使用图像样式的列表符号。

list-style-position 属性取值如表 9-10 所示。

**表 9-10　list-style-position 属性说明表**

| 属　性　值 | 说　　明 |
|---|---|
| outside | 默认值，将标志放在文本之外，而且任何换行文本在标志下均不对齐 |
| inside | 将标志放在文本之内，而且任何换行文本在标志下均对齐 |

**【例 9-5-1】** CSS 列表属性综合应用。代码如下所示，页面效果如图 9-8 所示。

视频讲解

```
<!-- edu_9_5_1.html -->
<!doctype html>
<html lang="en">
    <head>
        <meta charset="UTF-8">
        <title>CSS列表属性综合应用</title>
        <style type="text/css">
            h3{color:"#ffffff";background-color:#9999ff;text-align:center;}
            #li1{list-style-type:square;}
             #li2{list-style-type:upper-roman;}
             #li3{list-style-image:url("smallico1.bmp");list-style-
            position:inside;}
             #li4{list-style-image:url("smallico1.bmp");list-style-
             position:outside;}
             .sp1{font-weight:bolder;color:blue;}
         </style>
     </head>
     <body>
```

```
17          <h3>CSS列表属性综合应用</h3>
18          <ul id="li1">
19              <li>专业目录
20                  <ol id="li2">
21                      <li>计算机科学与技术专业</li>
22                      <li>软件工程</li>
23                      <li>信息管理与信息系统</li>
24                  </ol>
25              </li>
26              <li>图书
27                  <ul id="li3" >
28                      <li><span class="sp1">[inside]</span>计算机网络：计算机
                        网络所属现代词，指的是将地理位置不同的具有独立功能的多台计算机
                        及其外部设备，通过通信线路连接起来，在网络操作系统，网络管理软
                        件及网络通信协议的管理和协调下，实现资源共享和信息传递的计算机
                        系统。</li>
29                      <li id="li4"><span class="sp1">[outside]</span>数据库
                        原理：是数据库初学者和初级开发人员不可多得的数据库宝典，其中融
                        入了作者对数据库深入透彻的理解和丰富的实际操作经验。与第2版一
                        样，本版也深入浅出地描绘了数据库原理及其应用。</li>
30                  </ul>
31              </li>
32              <li>期刊目录</li>
33          </ul>
34      </body>
35  </html>
```

上述代码中定义了四个 id 样式，第 9 行定义列表样式类型为■；第 10 行定义列表样式类型为大写罗马字母；第 11 行定义用图像代替列表项符号并使用 inside 格式，文本环绕图像；第 12 行定义用图像代替列表项符号并使用 outside 格式，图像悬挂在文本的左边，并不环绕。

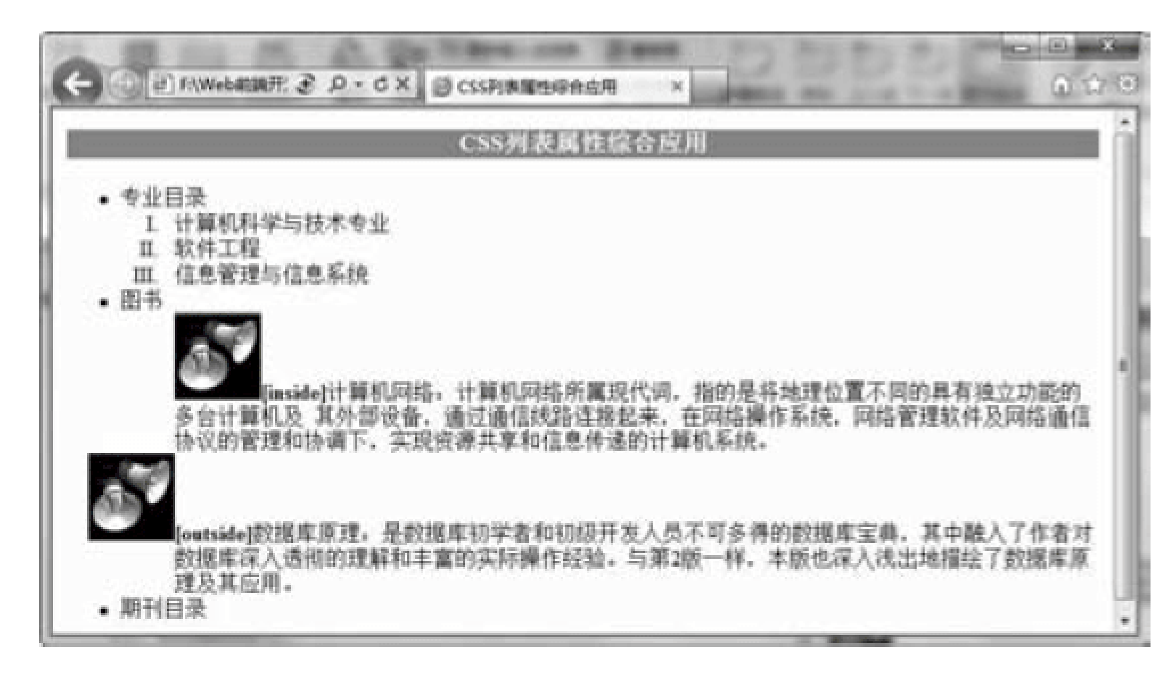



图 9-8　设置列表属性

# 9.6　CSS 盒模型

## 9.6.1　CSS 盒模型结构

在网页设计中，每个元素都是长方形的盒子，便产生了特定的盒子模型。在盒子模型中，重要的概念有边界（Margin）、边框（Border）、填充（Padding）、内容（Content），简

称为 MBPC 模型，如图 9-9 所示。边界又称为外边界（外补丁或外空白），是盒子边框与页面边界或其他盒子之间的距离。填充又称为内边界（内补丁或内空白），即内容与边框之间的距离。

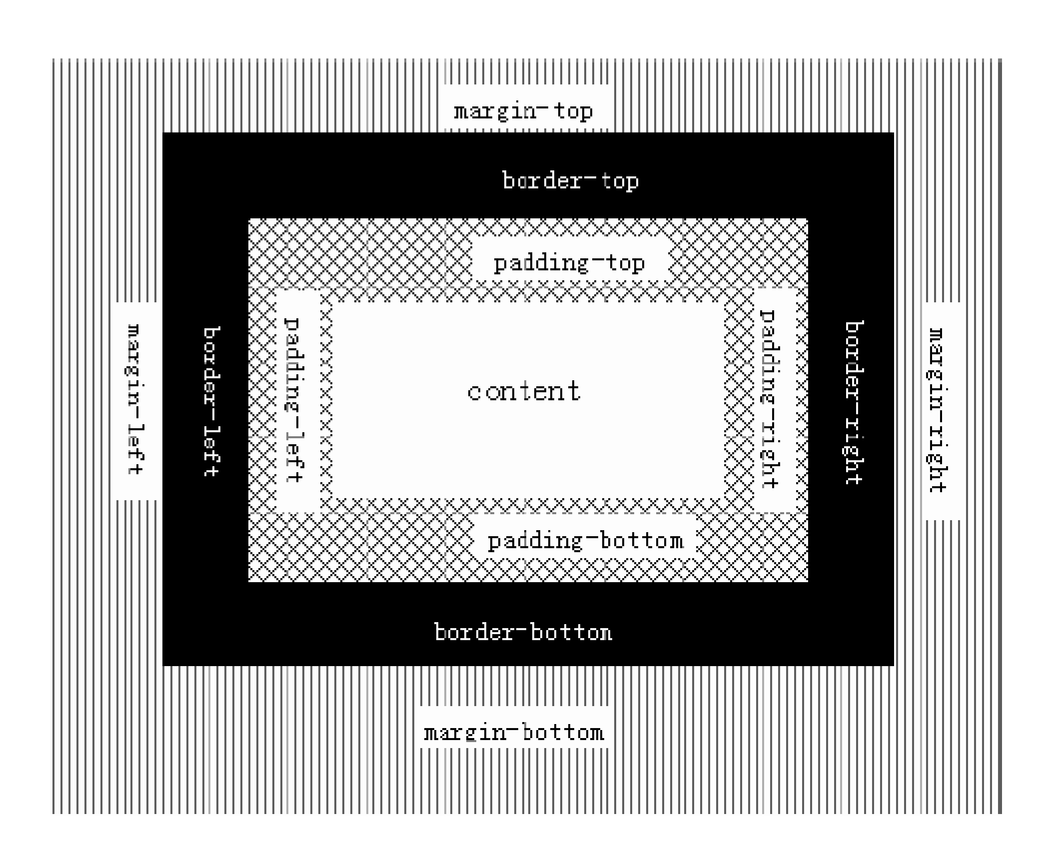



图 9-9　CSS 盒子模型

## 9.6.2　边界属性设置

边界属性是 margin，也称为外边距，表示盒子边框与页面边界或其他盒子之间的距离，属性值为长度值、百分数或 auto，属性设置的效果是围绕元素边框的“空白”。

**1. 基本语法**

```
margin-(top|right|bottom|left): 长度单位|百分比单位|auto
```

**2. 语法说明**

auto：表示采用默认值，浏览器计算边距。

长度单位和百分比单位：参考 9.1 节的介绍进行设置。

设置边界需要设置四个参数值，分别是表示“上、右、下、左”四个边。如果只设置一个参数值，则表示四个边界均相同。如果只设置两个参数值，那么第 1 个参数表示上、下边界值，第 2 个参数表示左、右边界。如果设置三个参数，那么第 1 个参数表示上边界，第 2 个参数表示左、右边界，第 3 个参数表示下边界。例如：

```
margin:10px 10px 20px 30px;/*分别设置上、右、下、左边界 */
margin:10px 20px 10px;      /*设置上边界为10px、左右边界为20px、下边界为10px*/
margin:20px 10px ;          /*设置上下边界为20px、左右边界为10px*/
margin:10px ;               /*设置4个边界均为10px */
p{margin-top:20px;}
p{margin-right:2em;}
h1{margin-bottom:30px;}
h3{margin-left:200%;}
```

**【例 9-6-1】**设置边界属性。代码如下所示，页面效果如图 9-10 所示。



视频讲解

```
<!-- edu_9_6_1.html -->
<!doctype html>
<html lang="en">
  <head>
```

```
5        <meta charset="UTF-8">
6        <title>设置边界属性</title>
7        <style type="text/css">
8            #p1{background:#99ffcc;margin-top:20px;margin-left:20px;}
9            #p2{background:#99ffff;margin:20px 30px 20px;}
10       </style>
11   </head>
12   <body>
13       <h4>设置边界属性</h4>
14       <p id="p1">使用CSS+DIV进行页面布局是一种全新的体验，完全有别于传统的表格
         排版习惯。</p>
15       <p id="p2">使用CSS+DIV进行页面布局是一种全新的体验，完全有别于传统的表格
         排版习惯。</p>
16   </body>
17 </html>0
```

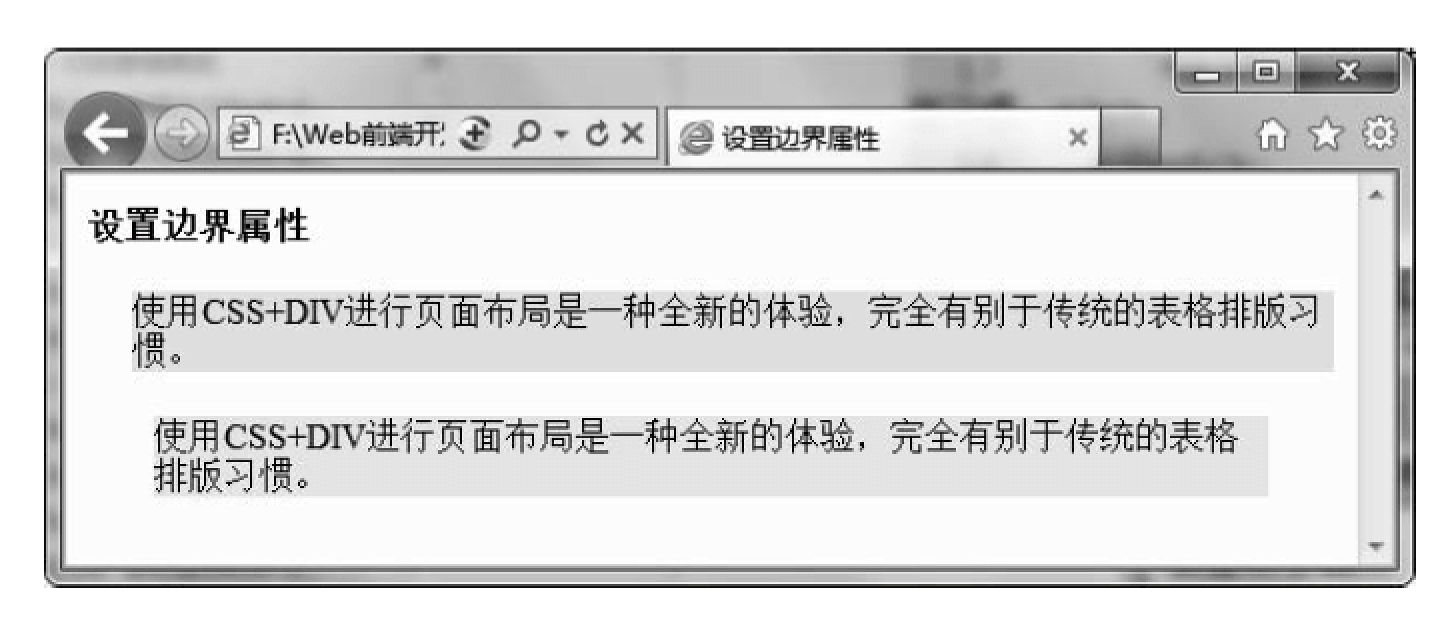



图 9-10　设置边界属性

**3. 代码解释**

代码中定义了两个 id 样式，第 8 行定义段落 1 背景颜色为#99ffcc、上边界为 20px、左边界为 20px；第 9 行定义段落 2 背景颜色为#99ffff，上边界为 20px，左、右边界为 30px，下边界为 20px。

## 9.6.3　边框属性设置

边框属性是 border，用于设置边框的宽度、样式以及颜色。

**1. 边框样式 border-style 属性**

border-style 属性用于设置不同风格的边框样式。

1）基本语法

```
border-style:none|hidden|dotted|dashed|solid|double | groove | ridge |
inset | outset
```

2）语法说明

语法中的属性值如表 9-11 所示。

**表 9-11　border-style 属性及说明表**

| 属　性　值 | 说　　明 |
|---|---|
| none | 定义无边框 |
| hidden | 与 none 相同。应用于表时例外，用于解决边框冲突 |
| dotted | 定义点状边框 |
| dashed | 定义虚线 |

续表

| 属　性　值 | 说　　明 |
|---|---|
| solid | 定义实线 |
| double | 定义双线。双线的宽度等于 border-width 的值 |
| groove | 定义 3D 凹槽边框。其效果取决于 border-color 的值 |
| ridge | 定义山脊状边框。其效果取决于 border-color 的值 |
| inset | 定义使页面沉入感边框。其效果取决于 border-color 的值 |
| outset | 定义使页面浮出感边框。其效果取决于 border-color 的值 |

与 margin 属性类似，border-style 属性可以设置多个值。比如下面的规则为类名为 cont 的段落定义了四种边框样式：实线上边框、点线右边框、虚线下边框和点线左边框。

```
p.cont{border-style: solid dotted dashed ;}
```

边框样式也可以通过单边样式属性进行设置，具有四个单边边框样式属性：

```
border-top-style：样式值;
border-right-style：样式值;
border-bottom-style：样式值;
border-left-style：样式值;
```

**2. 边框宽度 border-width 属性**

border-width 属性用于设置边框的宽度，其值可以是长度值或关键字 thin、medium、thick。

1）基本语法

```
border-width : medium(默认值) | thin | thick | length
```

2）语法说明

medium：默认宽度。thin：小于默认宽度。thick：大于默认宽度。length：请参考 9.1 节的介绍进行设置。

border-width 属性可以设置多个值，下面示例代码的效果是设置上边框和下边框为细边框、右边框和左边框为 10px。

```
border-width: thin 10px;
```

边框宽度也可以通过单边宽度属性进行设置，具有四个单边边框宽度属性：

```
border-top-width：样式值;
border-right-width：样式值;
border-bottom-width：样式值;
border-left-width：样式值;
```

**3．边框颜色 border-color 属性**

border-color 属性用于设置边框的颜色，与 color 类似。

border-color 属性可以设置多个值。

1）基本语法

```
border-color :color
```

2）语法说明

color 的值可以参考 9.4 节所给出的方法设置。边框颜色也可以通过单边颜色属性进行设置，具有四个单边边框颜色属性：

```
border-top-color：样式值；
border-right-color：样式值；
border-bottom-color：样式值；
border-left-color：样式值；
```

如果对上、下、左、右四条边框设置同样的样式、宽度、颜色，可以直接使用 border 属性，例如下面的示例代码为类名为 d2 的 div 设置了厚边框、实线、红色。

```
div.d2{border: thick solid red;}
```

**4．边框 border 复合属性**

边框 border 是一个复合属性，可以一次设置边框的粗细、样式和颜色。

1）基本语法

```
border: border-width | border-style | border-color
```

2）语法说明

该属性是复合属性。请参阅各参数对应的属性。

【例 9-6-2】设置边框属性。代码如下所示，页面效果如图 9-11 所示。

视频讲解

```
1 <!-- edu_9_6_2.html -->
2 <!doctype html>
3 <html lang="en">
4     <head>
5         <meta charset="UTF-8">
6         <title>设置边框</title>
7         <style type="text/css">
8             #p1{background:#99ffcc;border:15px groove #33ff66 ;}
9             #p2{border-style: dashed solid;}
10            #p3{border-style:solid;border-width:8px 10px;}
11            h4{text-align:center;padding:10px;background:#99cc99;}
12        </style>
13    </head>
14    <body>
15        <h4>设置边界</h4>
16        <p id="p1">使用CSS+DIV进行页面布局是一种全新的体验，完全有别于传统的表格
          排版习惯。</p>
17        <p id="p2">使用CSS+DIV进行页面布局是一种全新的体验，完全有别于传统的表格
          排版习惯。</p>
18        <p id="p3">使用CSS+DIV进行页面布局是一种全新的体验，完全有别于传统的表格
          排版习惯。</p>
19    </body>
20 </html>
```

3）代码解释

代码中定义段落的三个 id 样式，第 8 行定义段落 1 背景颜色为#99ffcc，采用 border 复合属性设置边框为粗细为 15px、线型为 groove、颜色为#33ff66；第 9 行定义段落 2 边框

样式上下边框为 dashed、左右边框为 solid；第 10 行定义段落 3 边框样式为实线型，上下边框为 8px、左右边框为 10px。

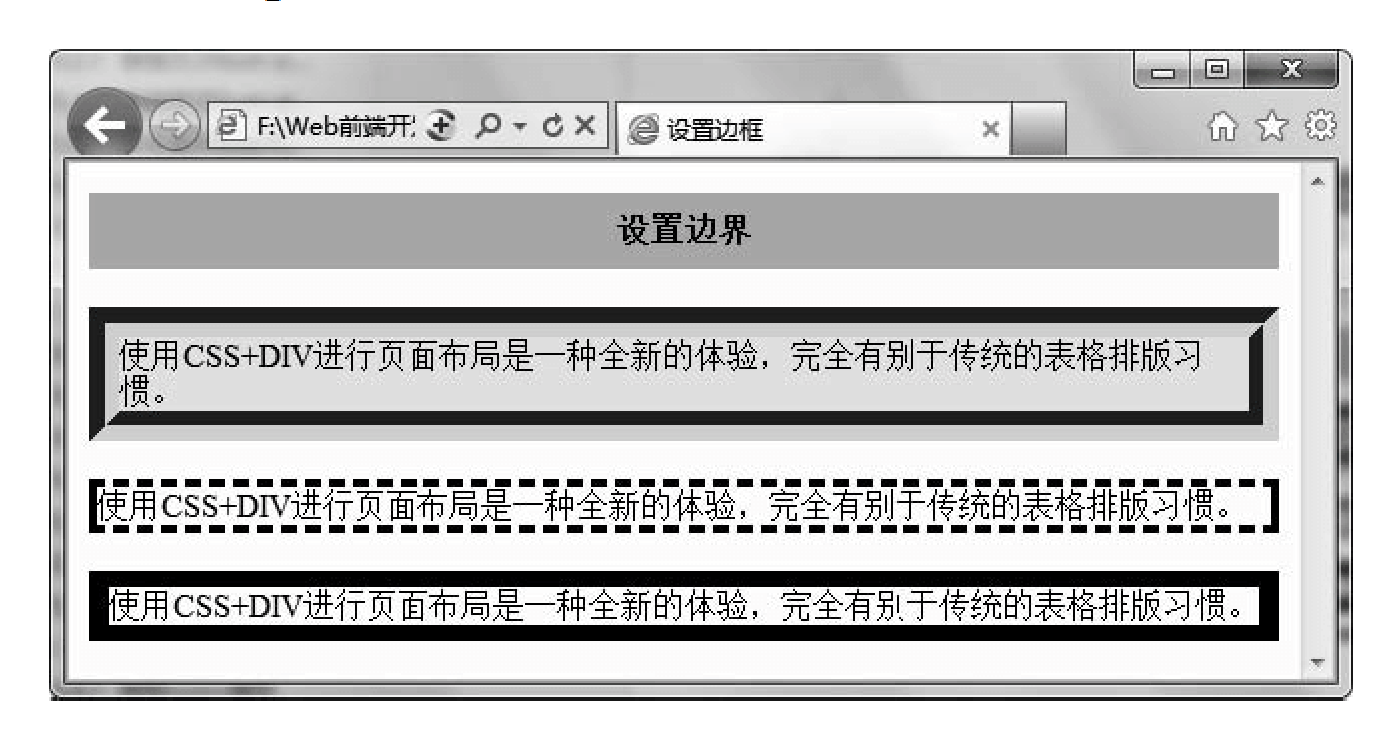



图 9-11　设置边框属性

## 9.6.4　填充属性设置

填充属性是 padding，也称为内边界，表示元素内容与边框之间的距离，属性值为长度值、百分数，属性设置的效果是包含在元素边框里面并围绕着元素内容的“元素背景”，也称内空白。

**1. 基本语法**

```
padding:长度|百分比
```

**2. 语法说明**

padding 属性可以为一个、二个、三个和四个值。表示方法同边框属性设置类似。

填充效果也可以通过单边填充属性进行设置，具有四个单边填充属性。

- padding-top：长度|百分比。
- padding-right：长度|百分比。
- padding-bottom：长度|百分比。
- padding-left：长度|百分比。

padding 属性值的设置如下所示：

```
h1 { padding-top:10px;          /* 分别表示上内边界*/
     padding-right:0.5em;       /* 分别表示右内边界*/
     padding-bottom:5px;        /* 分别表示下内边界*/
     padding-left:20%;          /* 分别表示左内边界*/        }
p{padding:10px 20px 30px 40px}  /* 分别表示上、右、下、左内边界*/
```

【例 9-6-3】设置填充属性。代码如下所示，页面效果如图 9-12 所示。



视频讲解

```
<!-- edu_9_6_3.html -->
<!doctype html>
<html lang="en">
   <head>
      <meta charset="UTF-8">
      <title>设置填充属性</title>
```

```
7        <style type="text/css">
8            #p1{background:#99ffcc;padding:15px 20px 15px;}
9            #p2{background:#99ff99;border-style:dashed;padding-top:20px;
             padding-bottom:20px;}
10           #p3{background:#99cccc;border-style:solid;padding-left:50px;
             padding-right:20px;}
11           h4{text-align:center;padding:10px;background:#99cc99;}
12       </style>
13   </head>
14   <body>
15       <h4>设置填充属性</h4>
16       <p id="p1">使用CSS+DIV进行页面布局是一种全新的体验，完全有别于传统的表格
         排版习惯。</p>
17       <p id="p2">使用CSS+DIV进行页面布局是一种全新的体验，完全有别于传统的表格
         排版习惯。</p>
18       <p id="p3">使用CSS+DIV进行页面布局是一种全新的体验，完全有别于传统的表格
         排版习惯。</p>
19   </body>
20 </html>
```

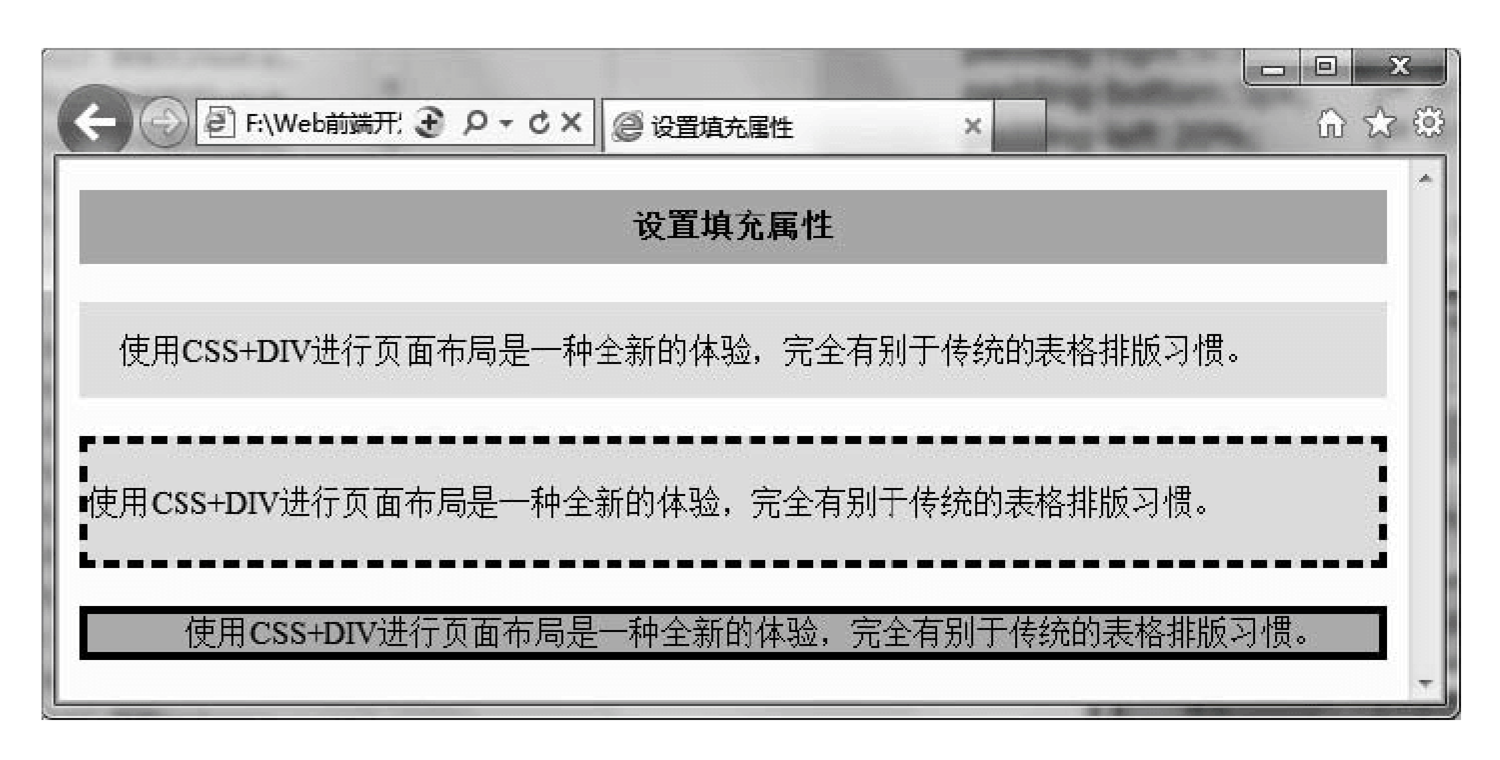



图 9-12　设置填充属性

**3. 代码解释**

代码中定义三个 id 样式，第 8 行定义段落 1 背景颜色为#99ffcc，采用 padding 复合属性设置内边界分别是上 15px、左右 20px、下 15px；第 9 行定义段落 2 背景颜色为#99ff99，并设置上内边界 20px、下内边界 20px；第 10 行定义段落 3 背景颜色为#99cccc，并设置左内边界 50px、右内边界 20px。

## 9.7　综 合 实 例

以上海美橙科技信息发展有限公司旗下网站——“建站之星”中提供的通用模板（模板号：2576）为例（http://sitestar.cndns.com/website/templates.aspx#），模仿设计“中国环宇科技有限公司”网站，如图 9-13 所示。采用 div 完成布局设计，编写相关 CSS 文件完成页面美化工作。

**1. 页面布局规划**

根据图 9-13 页面布局效果，很容易看出这是标准 5 行 2 列布局样式。使用布局绘图软

件画出布局图，如图 9-14 所示。




图 9-13　通用模板截图

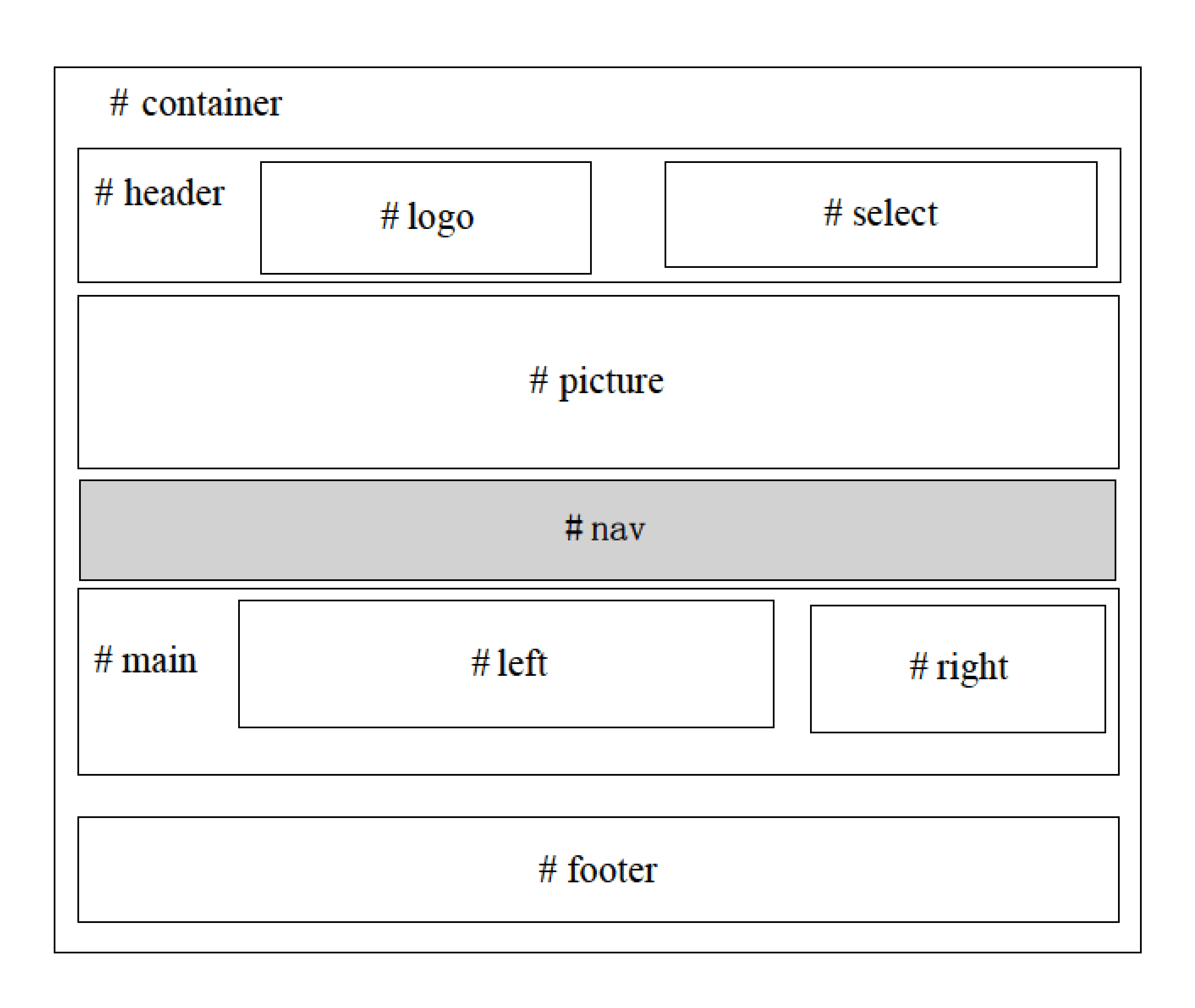



图 9-14　网站设计 div 布局

**2. 写出 div 结构代码**

新建一个 HTML 文档，编写如下的 div 嵌套结构代码。

```
<div id="container" class="">
   <div id="header" class="">
      <div id="logo" class=""></div>
      <div id="select" class=""></div>
   </div>
```

```
    <div id="picture" class=""></div>
    <div id="nav" class="">   </div>
    <div id="main" class="">
        <div id="left" class=""></div>
        <div id="right" class=""></div>
    </div>
    <div id="footer" class="">
    </div>
</div>
```

### 3．构造 huanyu.css 框架结构文件

根据 div 结构中的 ID 定义 CSS 文件中 id 样式，必须与 div 结构一一对应。

```
/* huanyu.css */
#container{}
#header{}
#logo{}
#picture{}
#main{}
#left{}
#right{}
#footer{}
```

### 4．编写 HTML 代码

```
<!-- edu_9_7_1.html -->
<!doctype html>
<html lang="en">
    <head>
        <meta charset="UTF-8">
        <title>中国环宇科技有限公司网站</title>
        <link type="text/css" rel="stylesheet" href="css/huanyu.css" >
    </head>
    <body>
        <div id="container" class="">
            <div id="header" class="">
                <div id="logo" class="">
                    <img src="images/logo.png" border="0" alt="">
                </div>
                <div id="select" class="">
                    <select name="" id="sel">
                        <option value="" selected>简体中文</option>
                        <option value="">繁体中文</option>
                        <option value="">ENGLISH</option>
                    </select>
                </div>
            </div>
            <div id="picture" class="">
        <img src="images/huanyu.jpg" width="990" height="345" border="0" alt="">
            </div>
            <div id="nav" class="">
                <table width="100%" height="40px" align="center" cellpadding="0px" cellspacing="0px">
                    <tr>
                        <td><a href="">首页</a></td>
                        <td><a href="">关于我们</a></td>
                        <td><a href="">服务项目</a></td>
```

```
                        <td><a href="">合作伙伴</a></td>
                        <td><a href="">联系我们</a></td>
                    </tr>
                </table>
            </div>
            <div id="main" class="">
                <div id="left" class="">
                    <h1>关于我们</h1>
                    <div id="left-up" class="">
                        <img src="images/xwjj.jpg" border="0" alt="">
                        <p>中国环宇科技有限公司是于1985年为了通过试验评价技术的
                          支援以提高产业技术而成立的试验评价机构，是和先进（发达）
                          国家的试验、认证机构进行交流和合作的某代表性机构。为了保
                          护本国产业的各种认证制度日渐完善，为保护消费者安全和环境
                          的各种制度的重要性日趋增加，KTL 为适应形势的发展，从产品
                          开发到获得认证的整个阶段提供支援，以帮助企业提高技术能力
                          以及拥有更强的竞争力。</p>
                    </div>
                    <div id="left-down" class="">
                        <img src="images/ico1.gif" border="0" alt="">
                        <p id="p1">我们的服务： 权威性的认证，严谨规范、完善周到
                        的服务，合理公平的费用，解除您在认证方面的后顾之忧。</p>
                    </div>
                </div>
                <div id="right" class="">
                    <h1>合作伙伴</h1>
                    <ul>
                        <li>XX代办服务公司</li>
                        <li>香港XX企业服务有限公司</li>
                        <li>上海XX专利代办机构</li>
                        <li>中国某某商业合作社</li>
                        <li>南京某某商业银行</li>
                        <li>日本XX会社</li>
                        <li>中国某某商业合作社</li>
                    </ul>
                </div>
            </div>
            <div id="footer" class="">
                <p>COPY RIGHT &copy;   中国环宇科技有限公司
                  科技事业部支持 京备XXXXX-342</p>
            </div>
        </div>
    </body>
</html>
```

### 5. 编写具有真实效果的 CSS 文件

```
/* huanyu.css */
*{font-size:12px;font-family:Times New Roman;}
#container{margin:0 auto;padding:0 auto;}
#header{width:990px;height:65px;margin:0 auto;}
#logo{width:263px; height:65px; float:left;
   background:url("images/logo.png") no-repeat left bottom;}
#select{width:727px; height:65px;
       float:left;text-align:right;}
#select #sel{margin-top:15px;}
```

```
#picture{width:990px;height:345px; clear:both;}
#nav{width:990px;height:40px;
   background:#0099CC; border:0px;}
a:link,a:visited,a:active{text-decoration:none;color:#FFFFFF;}
#nav a:hover{color:#333333;text-decoration:none;background:#F6F6F6;}
#nav a{width:194px;height:40px;}
td{line-height:40px;font-size:18px;text-align:center;vertical-align:
middle;}
#main{width:990px; height:250px; }
#left{width:660px;height:250px;
float:left;line-height:1.5em;}
h1{color:#0099FF;font-size:18px;
   height:36px; border-bottom:2px solid #0099FF;}
#left img{float:left;width:220px;height:144px;}
#left-down{margin:0px auto;padding:0px;
   clear:both;width:100%;height:70px;}
#left-down img{vertical-align:text-bottom;
   width:60px;height:50px;vertical-align:bottom;}
#p1{padding-top:20px;height:30px;}
#right{width:290px;  height:250px;
   float:right; border:1px solid #00ff00;}
ul{width:200px;height:100%;padding:0px;margin:0 auto;}
li{padding:0px;margin:0px;line-height:2em;
list-style-type:none;text-align:left;}
#footer{clear:both; width:990px;height:30px;
   background:#F7F7F7; border-top:2px solid #0099FF;}
#footer p{padding:10px auto;text-align:center;color:#333333;}
```

# 本 章 小 结

本章主要介绍了 CSS 的各种样式属性，包括文字样式、文本样式、颜色、背景、列表等。这些属性有的具有子属性，从不同方面描述外观样式，因而比较灵活，既可以使用单个子属性定义某一方面的样式，又可以使用复合属性定义整体的样式，在使用时应注意属性与属性之间的顺序及制约关系。

同时也重点介绍了 CSS 盒模型，它既是 CSS 的精华，也是学习的难点。如果把页面元素以“盒子”的方式呈现，那么便有了元素边界、元素边框、填充、元素内容这些重要概念。盒子具有四条边，所以这些属性都各有四个单边子属性，在使用时可以直接对某一条边应用单边子属性设置其样式，也可以按照一定顺序依次设置各边的样式，设置方式比较灵活。

# 练习与实验

## 练习 9

**1. 选择题**

（1）下列不属于 CSS 盒模型的属性是（　）。

（A）margin　　（B）padding　　（C）border　　（D）font

（2）边框的复合属性中不包括（　）。

（A）粗细　　　　（B）长短　　　　（C）颜色　　　　（D）样式

（3）下列可以去掉文本超链接的下画线的是（　）。

（A）a{text-decoration:no underline;}　（B）a{underline:none;}

（C）a{underline:false;}　　　　　　　（D）a{text-decoration:none;}

（4）下列不属于 CSS 文本对齐属性取值的是（　）。

（A）auto　　　　（B）left　　　　（C）center　　　　（D）right

（5）CSS 规则 p{margin:20px 10px;}的效果是（　）。

（A）仅设置了上边距为 20px，以及右边距为 10px

（B）仅设置了上边距为 20px，以及下边距为 10px

（C）设置了上、下边距为 20px，以及左、右边距为 10px

（D）设置了上、右边距为 20px，以及下、左边距为 10px

**2. 填空题**

（1）段落缩进的属性是______________；文本居中对齐的声明______________。

（2）实现背景图像在水平方向平铺的声明______________；设置背景图像位置的属性是______________。

（3）设置文字颜色为红色的声明（写出其值可能的所有形式）是 color:______________________________________________。

（4）声明“border: 2px double red; ”的含义是______________。

**3. 简答题**

（1）简述 CSS 盒模型概念。通过哪些属性可以描述一个具体的 CSS 盒模型？

（2）简述 CSS 列表样式属性及其取值情况。

## 实验 9

1．编写效果如图 9-15 所示的网页。网页中由左、右两个图层构成，左边 div 设置背景图像，图像居中显示，右边 div 设置背景图像填充效果，添加有效果文字内容。设计要求如下：

（1）HTML 中 div 结构如下：

```
<div id="wrap">
    <div id="pic"></div>
    <div id="text">
        <div id="title">木兰花令.拟古决绝词</div>
        <div id="author">纳兰性德</div>
        <div id="content">
            <p>人生若只如初见，</p>
            ...
        </div>
    </div>
</div>
```

（2）内容为“人生若只如初见，何事秋风悲画扇。等闲变却故人心，却道故心人易变。骊山雨罢清宵半，泪雨霖铃终不怨。何如薄幸锦衣郎，比翼连枝当日愿。”。

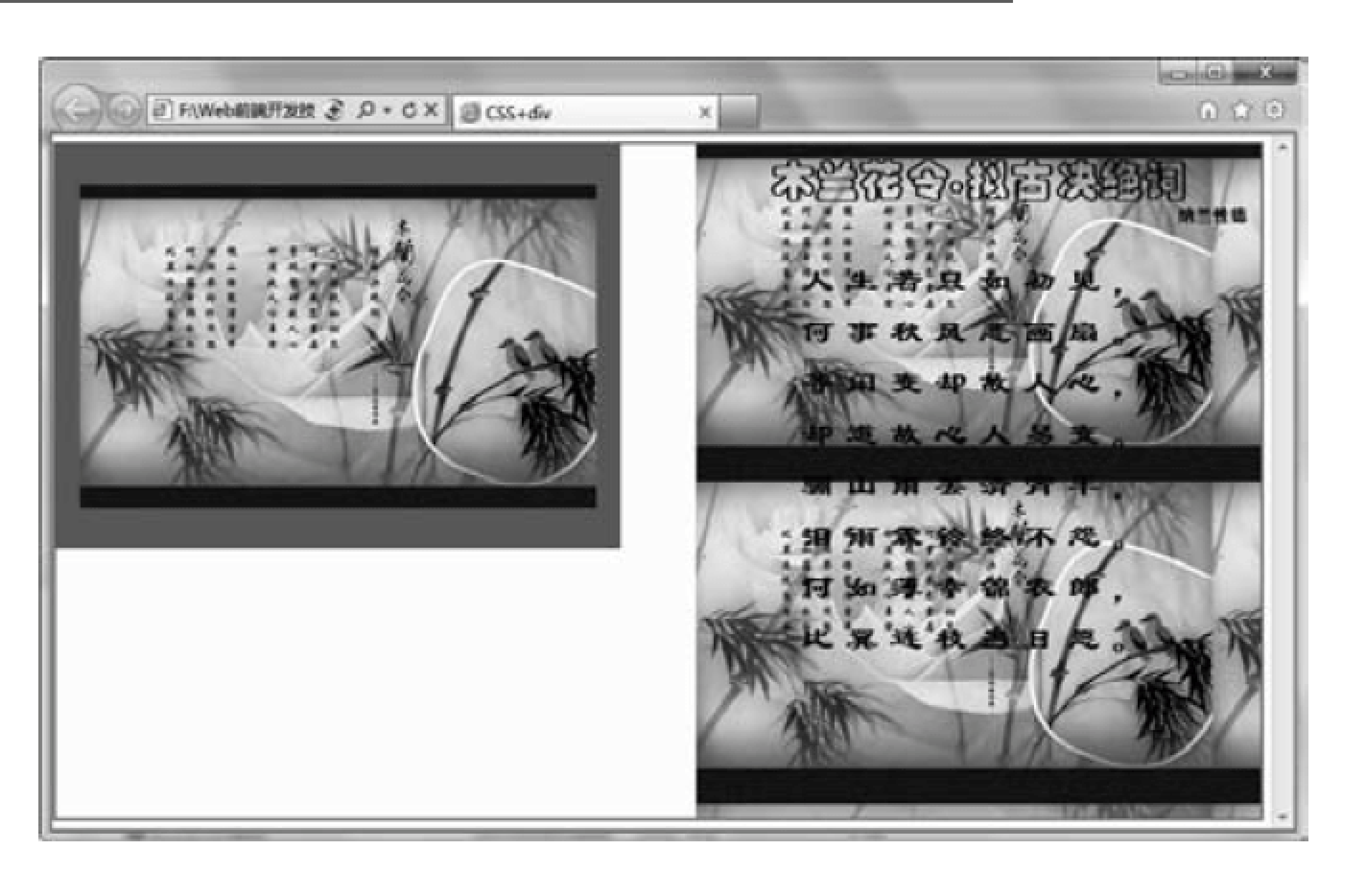


图 9-15　木兰花令效果图

（3）样式说明。

#wrap：宽度 900px、边界 0 auto、边框红色 2px 实线、上边界 5px。div：文本居中对齐。#pic：宽度 420px、高度 300px、背景图像为 ex8.jpg、不重复、位置居中、图像向左浮动、背景色为#77A。#text：背景图像为 ex8.jpg、向右浮动、宽度 420px、高度 500px、背景色为#77A、填充为 10px、字体粗细为 bold。#title：字体为“华文彩云”、大小为 32px。#author：字号大小为 12px、字体为黑体、文字右对齐、下边界为 24px。p：字体为隶书、字号大小为 24px、边界为 2px、字符间距为 0.5em、行高为 1.5em、文字居中对齐。

2．设计如图 9-16 所示的图文并茂的页面。设计要求：

（1）插入图像为 cup.jpg，图像向左浮动、边框为“1px 虚线、颜色为 gray”、边界为“10px 10px 10px 0”、填充为 5px。

（2）Mobile 首字母样式为“大小 3em、向左浮动”。

（3）h1 样式为“文字居中、白色、背景为#678”。

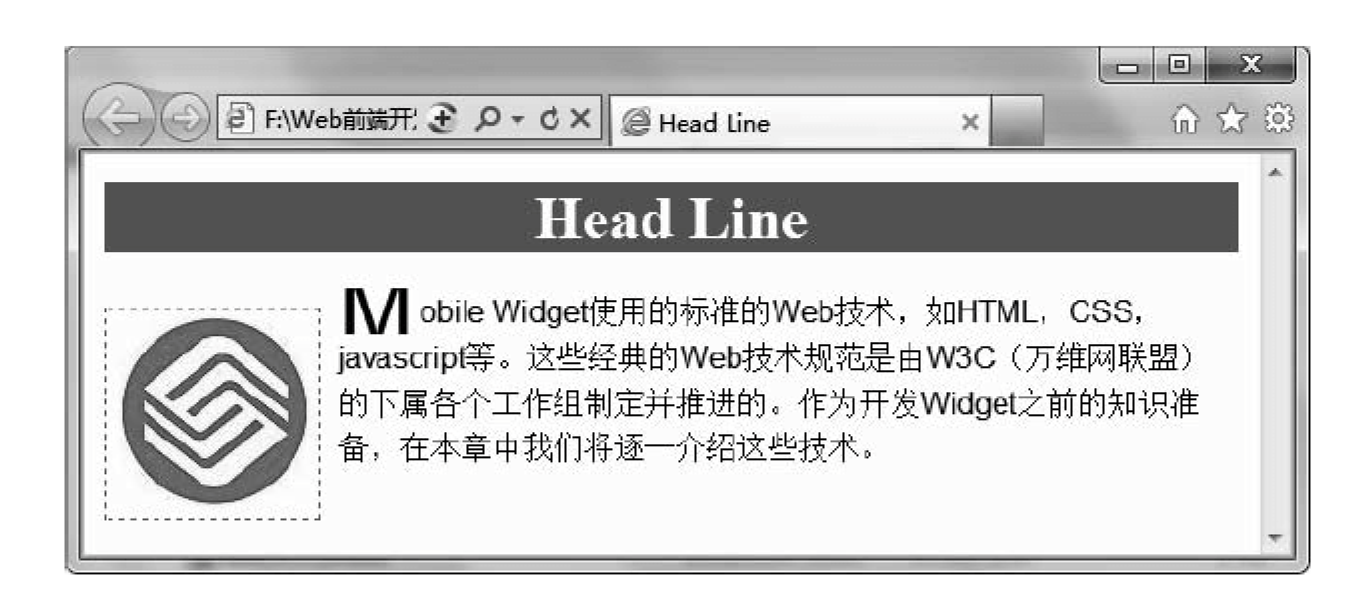



图 9-16　页面效果图

# 第 10 章　DIV+CSS 页面布局

**本章学习目标**

本章重点介绍使用 DIV+CSS 来规划各种页面布局方法、步骤以及 CSS 文件定义等，学会在不同的浏览器上进行页面效果的调试。

Web 前端开发工程师应掌握以下内容：

- 熟练地使用 DIV 标记的 CSS 各类属性。
- 掌握 CSS 定义与引用方法，学会使用外部样式表定义页面样式。
- 熟悉各类常见的页面布局类型，能够写出相应的 DIV 结构及 CSS 规则。
- 学会使用 DIV+CSS 进行页面布局，能够编写 HTML 代码和 CSS 文件。

## 10.1　页面布局设计

现在所有的主流的、大型的 IT 企业的网站布局几乎都采用 DIV、CSS 技术，有些甚至采用 DIV、CSS、表格混合进行页面布局。此类页面布局能够实现页面内容与表现的分离，提高网站访问速度、节省宽带、改善用户的体验。DIV+CSS 组合技术完全有别于传统的表格排版习惯。通过 DIV+CSS 实现页面元素精确控制，网站风格、代码维护与更新变得十分容易，甚至是页面的布局结构都可以通过修改 CSS 属性来重新定位。

DIV+CSS 布局的步骤大致为：首先整体上对页面进行分块，接着按照分块设计使用 div 标记，并理清 div 标记的嵌套和层叠关系，然后对各 div 标记进行 CSS 定位，最后在各个分块中添加相应的内容。

下面重点介绍常用的页面布局案例。

### 10.1.1　“三行模式”或“三列模式”

此模式特点是把整个页面水平、垂直分成三个区域，其中“三行模式”将页面分成头部、主体及页脚三部分；“三列模式”将页面分成左、中、右三个部分，如图 10-1 所示。

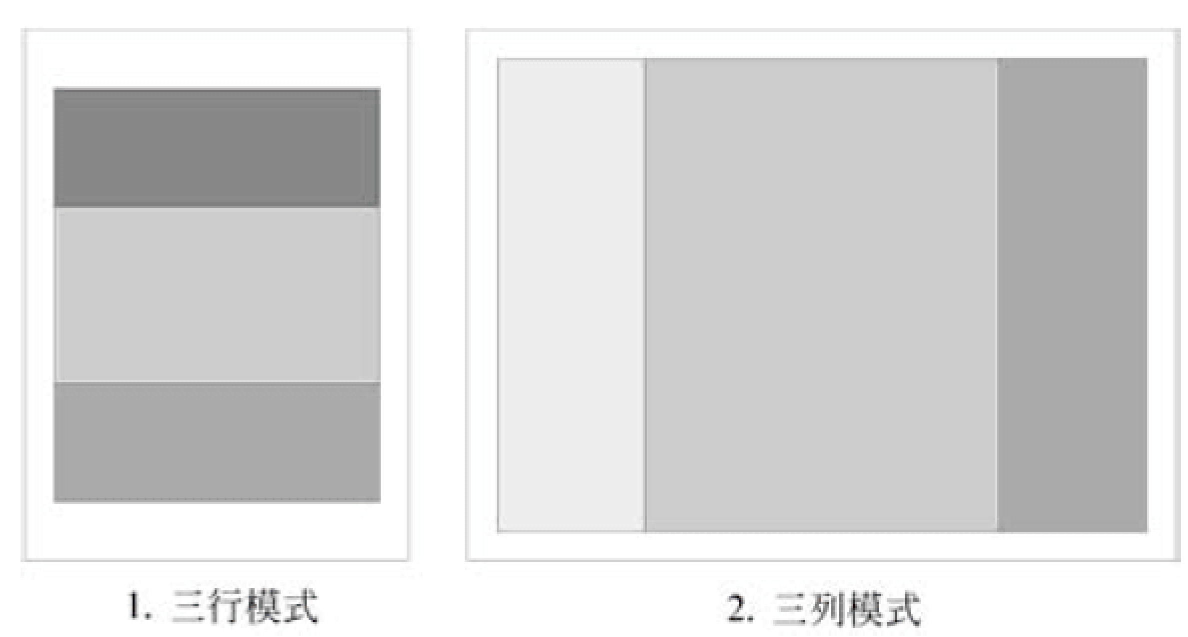



图 10-1　常用页面布局模式之一

根据页面布局情况，写出页面的 div 结构，两个模式 div 结构相似，具体代码如下：

```
<div id="header" class=""></div>
<div id="main" class=""></div>
<div id="footer" class=""></div>
```

然后编写相应的 CSS 文件，分别如下所示：

视频讲解

**1．三行模式**

```
/* layout1.css */
#header{width:100%;height:120px;background:#223344;}
#main{width:100%;height:500px;background:#553344;}
#footer{width:100%;height:40px;background:#993344;}
```



视频讲解

**2．三列模式**

```
/* layout2.css */
#left{width:30%;height:700px;background:#223344;float:left;}
#center{width:50%;height:700px;background:#553344; float:left;}
#right{width:20%;height:700px;background:#993344; float:left;}
```

## 10.1.2 “三行二列”“三行三列”模式

此模式特点是先将整个页面水平分成三个区域，再将中间区域分成两列或三列，如图 10-2 所示。

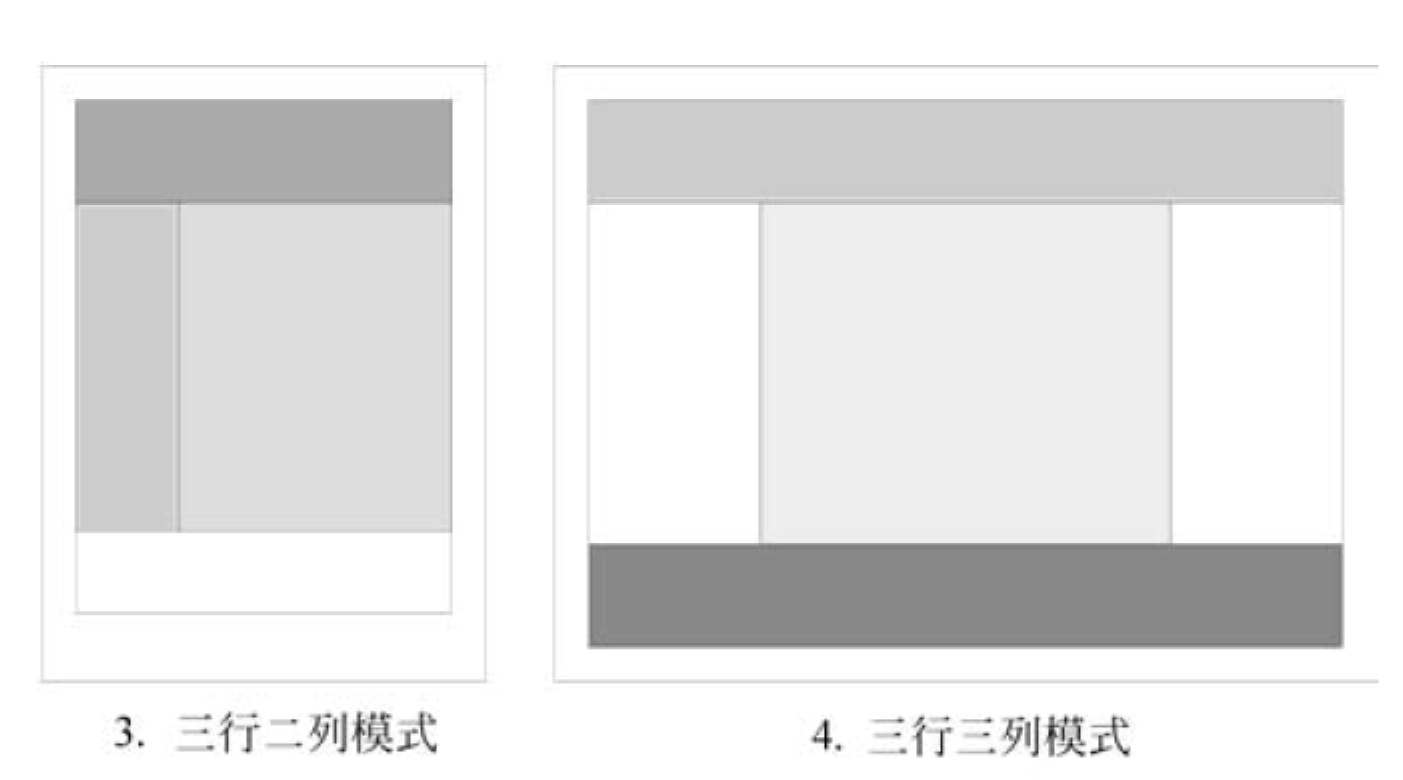



图 10-2　常用页面布局模式之二

对图 10-2 所示的页面进行布局 div 结构。两种模式的 div 结构分别如下：

- 三行二列模式的 div 结构。

```
<div id="header" class=""> header </div>
<div id="main" class="">
    <div id="left" class=""> left </div>
    <div id="right" class=""> right </div>
</div>
```

```
<div id="footer" class=""> footer </div>
```

- 三行三列模式的 div 结构。



视频讲解

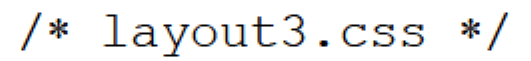
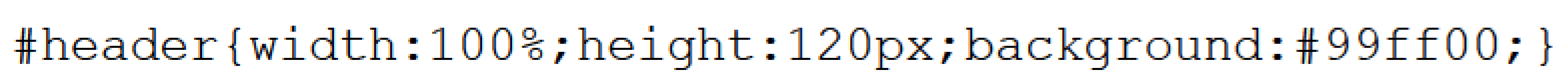
```
<div id="header" class=""> header </div>
<div id="main" class="">
    <div id="left" class=""> left </div>
    <div id="center" class=""> center </div>
    <div id="right" class=""> right </div>
</div>
<div id="footer" class=""> footer </div>
```

然后编写两种模式相应的 CSS 文件。

- 三行二列模式的 CSS 定义。



视频讲解

```
/* layout3.css */
#header{width:100%;height:120px;background:#99ff00;}
#main{width:100%;height:400px;background:#99ff99;}
#left{width:30%;height:100%;float:left;background:#999999;}
#right{width:70%;height:100%;float:left;background:#553344;}
#footer{clear:both;width:100%;height:80px;background:#66ff66;}
```

- 三行三列模式的 CSS 定义。

```
/* layout4.css */
#header{width:100%;height:120px;background:#99ff00;}
#main{width:100%;height:400px;background:#99ff99;}
#left{width:30%;height:100%;float:left;background:#999999;}
#center{width:40%;height:100%;float:left;background:#FF3344;}
#right{width:30%;height:100%;float:left;background:#553344;}
#footer{clear:both;width:100%;height:80px;background:#99ff66;}
```

在“三行三列模式”中，三列 div 可以同时向左、向右浮动，也可以左、中 div 向左、右 div 向右浮动或左 div 向左浮动，中、右 div 向右浮动。另外还可以左 div 向左浮动，右 div 向右浮动，中间 div 不浮动，而是设置填充 padding 属性来实现布局，只是中间 div（不浮动的 div）必须放在浮动 div 的后面才能生效，否则布局会混乱。

在实际使用 div 进行页面分块的过程中，需要注意的一个问题是，浮动的 div 的后续 div 中一定要先清除图层浮动，否则会影响其后 div 的显示效果。方法如下：

```
#div_n{clear:both|left|right;}
```

- 三列中的中间 div 不浮动时的 div 结构。

```
<div id="header" class=""> header </div>
<div id="main" class="">
    <div id="left" class=""> left </div>        <!-- 浮动的div -->
    <div id="right" class=""> right </div>     <!-- 浮动的div -->
```

```
    <div id="center" class=""> center </div>   <!-- 不浮动的div -->
</div>
<div id="footer" class=""> footer </div>
```

- 三列中的中间 div 不浮动时的 CSS 文件定义。

```
/* layout4_1.css */
#header{width:100%;height:120px;background:#99ff00;}
#main{width:100%;height:400px;background:#99ff99;}
#left{width:30%;height:100%;float:left;background:#999999;}
#center{padding:0px 30%;height:100%;background:#FF3344;} /*不浮动div*/
#right{width:30%;height:100%;float:right;background:#553344;}
#footer{clear:both;width:100%;height:80px;background:#99ff66;}
```

### 10.1.3　多行多列复杂模式

国内大型商业网站基本上多行多列模式布局，如图 10-3 所示。例如中央人民政府、中关村在线、淘宝网、腾讯、网易、新浪、搜狐、人民网等网站采用“多行三列模式”。公安部、财政部、阿里巴巴、网上超市 1 号店、去哪儿网、赶集网等网站采用“多行四列模式”。其他大多数网站布局根据首页的长度变化而略有差异，在此不再一一叙述。

根据图 10-3 所示进行页面布局设计。此处仅对“多行三列模式”的页面布局进行 div 结构划分，对“多行四列模式”读者可以自行模仿写出 div 结构。

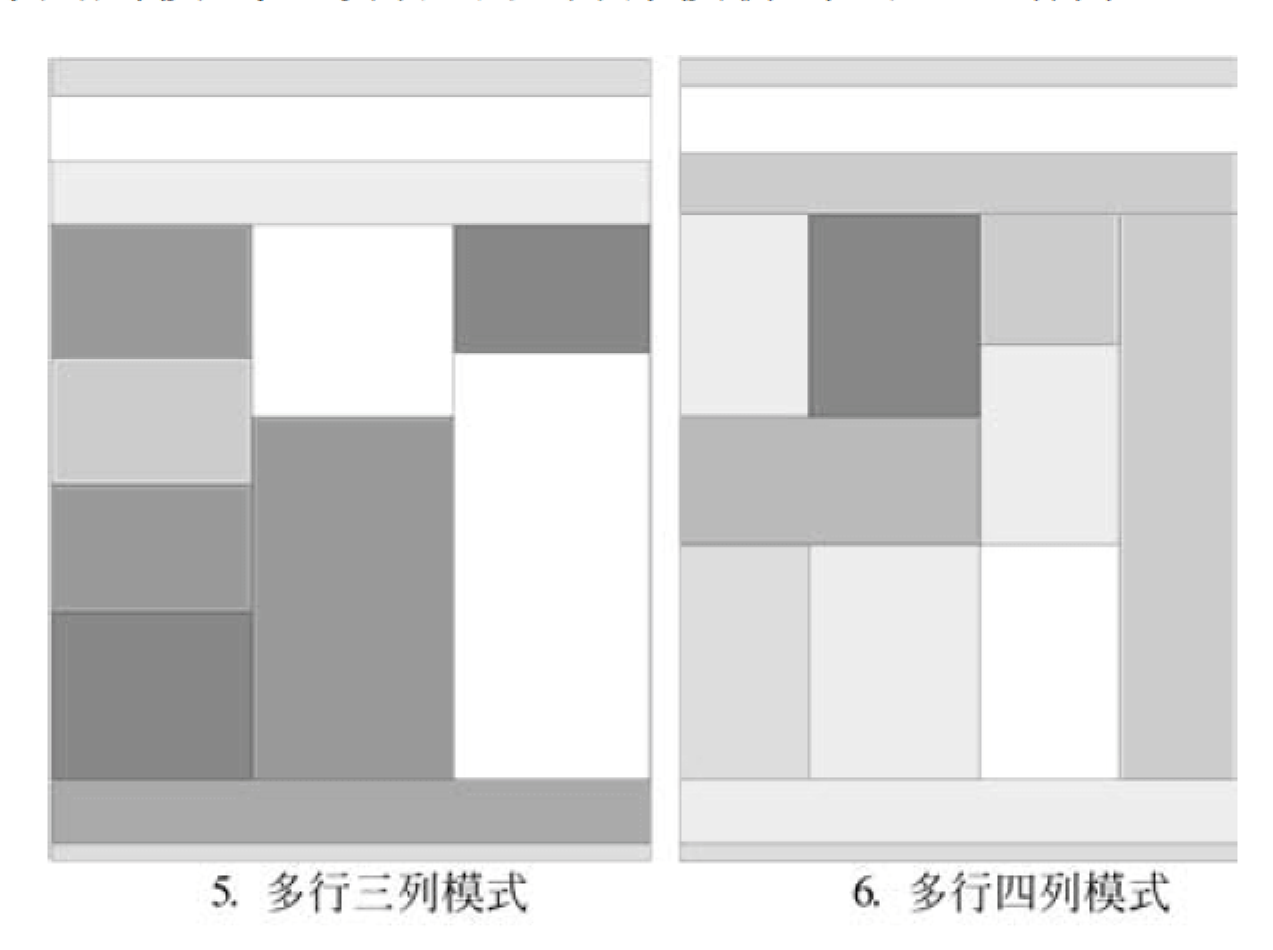



图 10-3　多行多列复杂模式

- 多行三列模式的 div 结构。

视频讲解

```
<div id="container" class="">
    <div id="header" class="">
        <div id="logo" class="">logo</div>
        <div id="nav" class="">nav</div>
    </div>
    <div id="main" class="">
        <div id="left" class="">
            <div id="left_up_1" class="">left_up_1</div>
            <div id="left_up_2" class="">left_up_2</div>
            <div id="left_down_1" class="">left_down_1</div>
```

```
            <div id="left_down_2" class="">left_down_2</div>
        </div>
        <div id="center" class="">
            <div id="center_up" class="">center_up</div>
            <div id="center_down" class="">center_down</div>
        </div>
        <div id="right" class="">
            <div id="right_up" class="">right_up</div>
            <div id="right_down" class="">right_down</div>
        </div>
    </div>
    <div id="footer" class="">footer</div>
</div>
```

- 多行三列模式的 CSS 定义。

```
/* layout5.css */
*{font-size:16px;margin:0 auto;padding:0px;}
#container{background:#334455;width:100%;height:700px;}
#header{background:#FF4455;width:100%;height:150px;}
#logo{background:#FFDD55;width:100%;height:100px;}
#nav{background:#FFDD99;width:100%;height:50px;}
#main{background:#33DD55;width:100%;height:500px;}
#left{background:#33FBFB;width:33%;height:100%;float:left;}
#left_up_1{background:#99BBDD;width:100%;height:125px;}
#left_up_2{background:#AABBCC;width:100%;height:125px;}
#left_down_1{background:#BBCCDD;width:100%;height:125px;}
#left_down_2{background:#CCDDEE;width:100%;height:125px;}
#center{background:#88FBFB;width:34%;height:100%;float:left;}
#center_up{background:#66ff66;width:100%;height:200px;}
#center_down{background:#45DD22;width:100%;height:300px;}
#right{background:#DDFBFB;width:33%;height:100%;float:left;}
#right_up{background:#55DDFB;width:100%;height:150px;}
#right_down{background:#667733;width:100%;height:350px;}
#footer{background:#DDDD11;width:100%;height:50px;}
```

在 HTML 代码中链接外部样式表 layout5.css，并在浏览器中打开 edu_10_1_5.html 页面，效果如图 10-4 所示。




图 10-4 多行三列布局效果图

# 10.2 导航菜单设计

导航菜单是网站重要的组成部分。导航菜单的设计关系着网站的可用性和用户体验，有吸引力的导航能够吸引用户去浏览更多的网站内容。设计一个优秀的页面导航菜单会给网站增色不少。作为一名 Web 前端开发工程师必须掌握传统的网站导航菜单设计技巧，同时也需要学习响应式导航菜单的设计方法。

网站菜单表现形式丰富多样。从菜单层次上看，可以分为一级、二级和多级菜单。从排列方式上看，可分为水平导航、垂直导航菜单。从技术实现角度上看，导航菜单通常采用无序列表、表格、超链接和样式表相结合的方法来实现，也可以使用如 CSS3 Menu、jQuery 等第三方插件技术来实现。

## 10.2.1 一级水平导航菜单

**1．采用“表格+超链接”来设计**

使用表格布局设计一级导航菜单非常容易而且布局均匀，根据导航栏目数量确定表格的列数。采用 1 行 10 列表格，第 1、第 10 单元格插入空格，留出左右边空白，其余单元格内插入超链接即可实现，代码如下所示：

```
 1 <table>
 2     <tr>
 3         <td> </td>
 4         <td><a href="#">首页</a> </td>
 5         <td><a href="#">期刊介绍</a></td>
 6         <td><a href="#">编委会/董事会</a></td>
 7         <td><a href="#">常见问题及解答</a></td>
 8         <td><a href="#">常用文档下载</a></td>
 9         <td><a href="#">订阅</a></td>
10         <td><a href="#">过刊浏览</a></td>
11         <td><a href="#">优先出版</a> </td>
12         <td> </td>
13     </tr>
14 </table>
```

上述代码中第 3 行、第 12 行单元格是插入空格，第 4～11 行单元格是利用超链接定义导航菜单。

对超链接和表格定义样式如下：

```
table{width:978px;height:40px;text-align:center;background:url("nav_blue.jpg");}
a:link,a:visited,a:hover,a:active{text-decoration:none;color:#FFFFFF;}
a:hover{color:red;border-bottom:5px solid #FF0000;}
```

应用上述 CSS 样式后导航菜单样式如图 10-5 所示。

**2．采用“无序列表+超链接”来设计**

采用无序列表设计“一级水平导航菜单”需要做两件事：一是要去掉列表项前面的符号；二是将垂直显示的列表项转换成水平显示。




图 10-5　采用表格和超链接制作导航菜单效果图

以“计算机应用研究”杂志网站的导航为例，采用无序列表设计期刊网站的导航菜单，其实现的 HTML 代码如下所示：

```
<div id="nav" class="">
    <div class="navwrap">
        <ul>
            <li><a href="/">首页</a></li>
            <li><a href="/html/intro.html">期刊介绍</a></li>
            <li><a href="/html/editorial_board.html">编委会/董事会</a></li>
            <li><a href="/html/faq.html">常见问题及解答</a></li>
            <li><a href="/html/downloads.html">常用文档下载</a></li>
            <li><a href="/html/subscribe.html">订阅</a></li>
            <li><a href="/article/01-index.html">过刊浏览</a></li>
            <li><a href="/article/02-index.html">优先出版</a></li>
        </ul>
    </div>
</div>
```

对无序列表、列表项分别定义如下的 CSS 样式后，导航菜单已由默认垂直排列状态改为水平排列方式，列表项前面没有符号，如图 10-6 所示。

```
/* 计算机应用研究杂志网站导航CSS */
#nav {width: 100%;font-size: 12px;
 background: #004183 url("nav_blue.jpg") top center repeat-x;}
.navwrap {width: 978px; height: 40px; margin: 0 auto;
 background: url("nav_blue.jpg") top center repeat-x;/* 设置背景图像 */}
ul{width: 898px;height: 40px;margin: 0;padding: 0 0 0 130px;
 list-style: none;/* 去除列表项前的符号 */}
li{float: left;/*设置列表项浮动  */}
a{line-height: 40px;font-weight: bold;
 margin: 0 10px;color: #fff;text-decoration: none;}
a:hover {color: #ff3d3d;}
```




图 10-6　采用无序列表和超接制作导航菜单效果图

垂直一级菜单实现起来比较容易。因为列表项默认就是以垂直方式显示的，所以不再考虑如何控制列表项，整体控制起来比较容易，采用表格和超链接、无序列表和超链接的方式均可以实现，此处不再赘述。

## 10.2.2　二级水平导航菜单

商业网站上导航菜单一般有多种表现形式，分别是一级导航菜单、二级导航菜单、多种形式并存的导航菜单。例如“淘宝论坛”（http://bbs.taobao.com/）和“京东网上商场”（http://www.jd.com/）主页就是采用多种形式并存的菜单的网站案例，如图 10-7 和图 10-8 所示。多级导航菜单的实现技术与二级导航菜单类似。




图 10-7　淘宝论坛首页导航菜单效果图




图 10-8　京东网上商场首页导航菜单效果图

### 1．下拉导航菜单

借助于 JavaScript 设计网站下拉菜单的案例比较多见，而采用纯 CSS 设计网站下拉菜单需要对样式进行详细定义才能实现。不过要考虑到不同浏览器之间的兼容性。下面列举一个仅仅采用<ul></ul>、<li></li>、<a></a>等标记和 CSS 样式定义来实现简单的二级下拉菜单的设计过程，页面效果如图 10-9 所示。




图 10-9　下拉导航菜单效果图

具体设计步骤如下：

（1）首先编写下拉菜单的 HTML 代码，链接外部样式表，代码如下所示。



视频讲解

```
<!-- edu_10_2_6.html -->
<!doctype html>
<html lang="en">
    <head>
        <meta charset="UTF-8">
        <title> 下拉导航菜单</title>
        <link rel="stylesheet" href="drapdownmenu.css" type="text/css">
    </head>
    <body>
        <ul>
            <li><a href="#">首页</a></li>
            <li><a href="#">jQuery特效</a>
                <ul>
                    <li><a href="#">jQuery图片特效</a></li>
                    <li><a href="#">jQuery导航特效</a></li>
                    <li><a href="#">jQuery选项卡特效</a></li>
                    <li><a href="#">jQuery文字特效</a></li>
                </ul>
            </li>
            <li><a href="#">JavaScript特效</a></li>
            <li><a href="#">Flash特效</a>
                <ul>
                    <li><a href="#">Flash图片特效</a></li>
                    <li><a href="#">Flash导航特效</a></li>
                    <li><a href="#">Flash选项卡特效</a></li>
                    <li><a href="#">Flash文字特效</a></li>
                </ul>
            </li>
            <li><a href="#">div+css教程</a></li>
            <li><a href="#">HTML5教程</a></li>
        </ul>
    </body>
</html>
```

在不设置任何 CSS 类的情况下，下拉菜单的页面效果如图 10-10 所示。




图 10-10　无样式的下拉导航菜单效果图

（2）逐步设置样式，让菜单越来越美。

① 定义 ul 的样式，设置边距和填充均为 0px。

```
ul {margin: 0px; padding: 0px;}  /*考虑到不同浏览器兼容性*/
```

② 定义列表项样式，由垂直排列改为水平排列。应用后页面效果如图 10-11 所示。

```
ul li { height: 30px; width: 115px; list-style: none; float: left;
display: inline;  font: 0.9em Arial, Helvetica, sans-serif;}
```

这条规则定义了 li 标记为浮动、行内显示、宽度、高度、字体等样式。




图 10-11　应用规则后的下拉导航菜单效果图

③ 定义超链接的样式，应用规则后的页面效果如图 10-12 所示。

```
ul li a { color: #FFF; width: 113px; margin: 0px; padding: 0px 0px 0px 8px;
          text-decoration: none; display: block;  background: #808080;
line-height: 29px; border-right: 1px solid #ccc; border-bottom: 1px solid #ccc;}
```

这条规则的作用就是加上背景和菜单间的隔离线，把默认有下画线蓝色的文字变成白色无下画线。




图 10-12　应用规则后的下拉导航菜单效果图

④ 定义嵌套列表项和子菜单超链接的规则。

```
ul li ul li { height:25px; }
ul li ul li a {background: #666; line-height:24px;} /*#666等同于#666666*/
```

此处第 1 条是设置子菜单的列表项目高度为 25px，以区别主菜单列表项；第 2 条规则是子菜单项中的超链接的背景改为#666，并将行高调整为 24px。应用样式后的页面效果如图 10-13 所示。

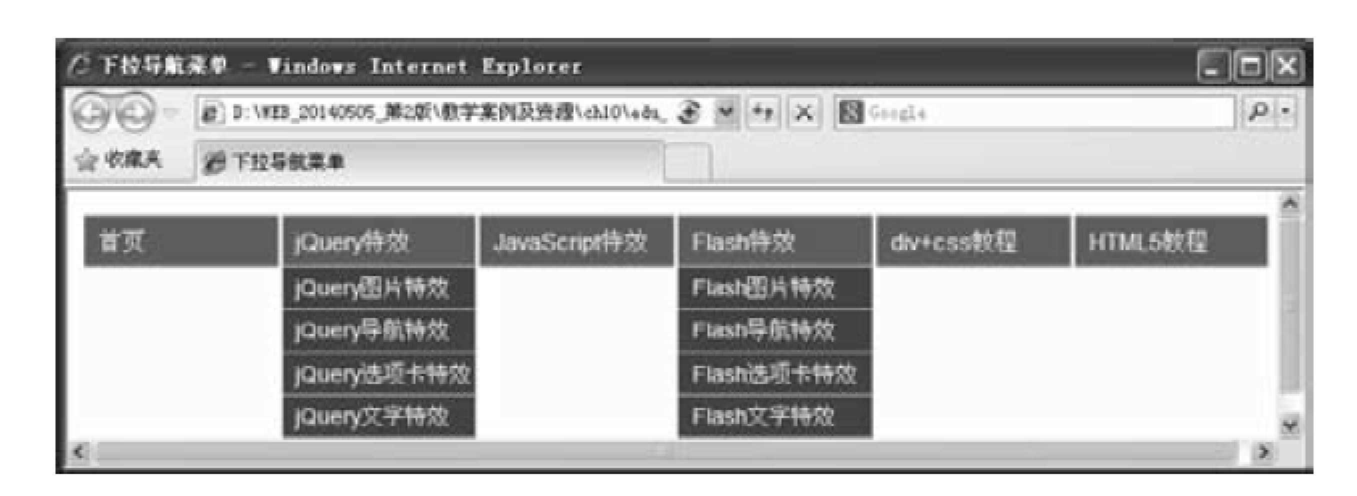



图 10-13　应用规则后的下拉导航菜单效果图

⑤ 定义鼠标滑过某个菜单项时的样式。

```
ul li a:hover { background: #666; border-bottom:1px dashed #FF0000; }
```

此处定义了鼠标滑过时背景色和子菜单的背景色一样，定义底边框为 1px、点划线、红色，页面效果如图 10-14 所示。




图 10-14　应用规则后的下拉导航菜单效果图

⑥ 定义子菜单项初始状态为隐藏，页面效果如图 10-15 所示。

```
ul li ul { visibility: hidden; }    /*也可以设置 display:none */
```




图 10-15　应用规则后的下拉导航菜单效果图

⑦ 定义鼠标滑过时下拉子菜单显示样式，页面效果如图 10-16 所示。

```
ul li:hover ul { visibility: visible; }   /*也可以设置 display:block */
ul li ul li a:hover { background: #333; }  /*#333等同于#333333*/
```

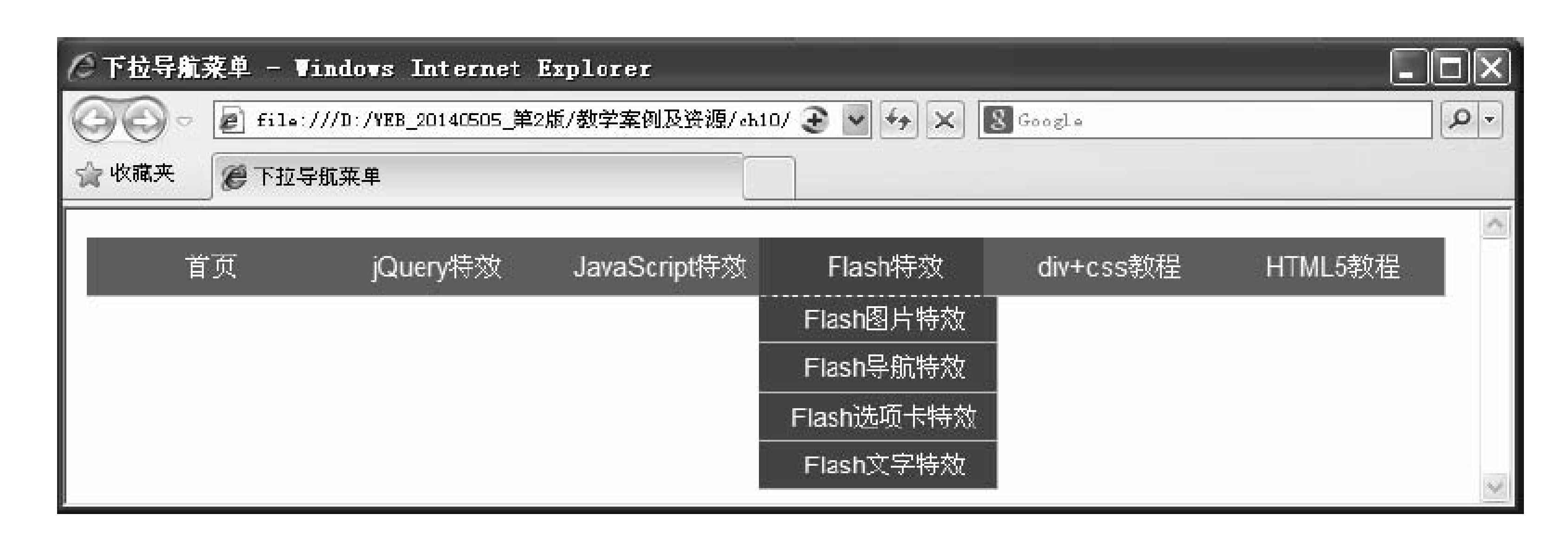



图 10-16　应用规则后的下拉导航菜单效果图

**2．横向二级导航菜单**

所谓横向二级导航菜单，就是一层主菜单是水平排列、二层子菜单也是水平排列，各占一行，其中二层子菜单可能会占多行，取决于子菜单的数量。例如“携程旅行网官网”(http://www.ctrip.com/)，如图 10-17 所示。




二级水平导航菜单

图 10-17　携程旅行网官网首页导航菜单效果图

采用纯 CSS 打造横向二级导航菜单，需要对 HTML 中的 div、ul、li、a 等标记进行样式定义，并应用样式。在设计下拉菜单的基础上，很容易实现横向二级导航菜单，如图 10-18 所示。

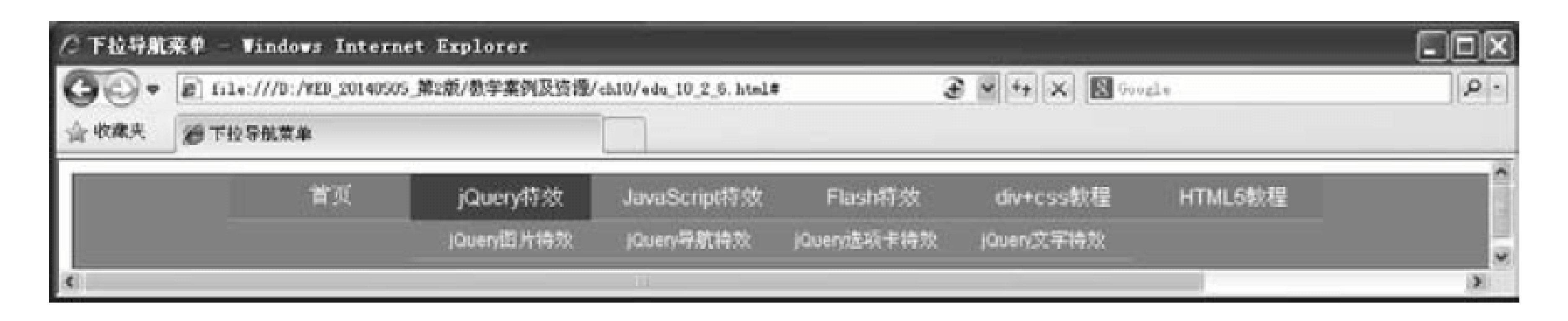



图 10-18　横向二级导航菜单效果图

具体设计步骤如下：

（1）设计 HTML 代码，与下拉菜单基本相似，代码如下所示。

```
<!-- edu_10_2_7.html -->
<!doctype html>
<html lang="en">
    <head>
        <meta charset="UTF-8">
        <title> 横向二级导航菜单</title>
        <link rel="stylesheet" href="level2_menu.css" type="text/css">
    </head>
    <body>
        <div id="menu" class="">
            <ul>
                <li><a href="#">首页</a></li>
                <li><a href="#">jQuery特效</a>
                    <div id="submenu" class="">
                        <ul>
                            <li><a href="#">jQuery图片特效</a></li>
                            <li><a href="#">jQuery导航特效</a></li>
```

```
              <li><a href="#">jQuery选项卡特效</a></li>
              <li><a href="#">jQuery文字特效</a></li>
          </ul>
        </div>
        </li>
        <li><a href="#">JavaScript特效</a></li>
        <li><a href="#">Flash特效</a>
          <div id="submenu" class="">
            <ul>
              <li><a href="#">Flash图片特效</a></li>
              <li><a href="#">Flash导航特效</a></li>
              <li><a href="#">Flash选项卡特效</a></li>
              <li><a href="#">Flash文字特效</a></li>
            </ul>
          </div>
        </li>
        <li><a href="#">div+css教程</a></li>
        <li><a href="#">HTML5教程</a></li>
      </ul>
    </div>
  </body>
</html>
```

与下拉菜单的不同之处在于二级导航子菜单是放在 div 中，id 为 submenu，需要定义子菜单图层 div 的样式。

（2）定义 HTML 中相关标记的样式。

```
/* 程序名称：level2_menu.css
   作用对象：edu_10_2_6.html*/
#menu{  /* 定义外层图层样式*/
  padding-left: 100px;
  margin: 0 auto;
  text-align: center;
  width: 100%;
  height: 60px;
  background: #55AAEE;
  border: 1px solid #333333;
}
#menu ul{/*考虑到不同浏览器兼容性*/
  margin: 0px;
  padding: 0px;
}
#submenu{  /*定义存放子菜单的图层样式*/
  width: 900px;  /* 不要为100% */
  height: 28px;
  text-align: center;
}
#menu ul li {  /*定义主菜单样式*/
  height: 30px;
  width: 115px;
  list-style: none;  /*去除列表项符号*/
  float: left;      /*列表项向左浮动 */
  display: inline;   /*列表项为行内显示*/
  font: 0.9em Arial, Helvetica, sans-serif;
  text-align: center;
```

```
}
ul li a {      /*定义主菜单中超链接样式*/
  color: #FFF;
  width: 114px;
  margin: 0px;
  padding: 0px 0px 0px 8px;
  text-decoration: none;
  display: block;      /*超链接以块方式显示 */
  background: #55A0FF;
  line-height: 29px;
  border-bottom: 1px solid #ccc;
}
ul li #submenu ul li {  /*定义子菜单中列表项的高度，与主菜单不同 */
  height: 25px;
}
ul li #submenu ul li a {  /*定义子菜单中超链接的样式 */
  background: #55AAEE;
  line-height: 24px;
}
ul li a:hover{  /*定义主菜单鼠标滑过时的样式 */
  background: #666;
  border-bottom: 1px dashed #FF0000;
}
ul li #submenu{  /*定义子菜单初始状态为不显示 */
  display: none;/*visibility: hidden;*/
}
#submenu ul li{   /*定义子菜单中列表项的样式 */
  height: 24px;
  width: 113px;
  list-style: none;
  float: left;
  display: inline;
  font: 0.8em Arial, Helvetica, sans-serif;
  text-align: center;
}
ul li:hover #submenu{  /* 主菜单鼠标滑过时显示子菜单*/
  display: block;/*visibility: visible;*/
}
ul li #submenu ul li a:hover{
  background: #333;/* 子菜单鼠标滑过时指定新的背景颜色*/
}
```

参照下拉菜单中 CSS 规则的定义，很容易写出横向二级导航菜单的样式文件。目前商业网站中的导航菜单大多数是采用 DIV+CSS+JavaScript 技术或采用 DIV+CSS+jQuery 技术来设计响应式导航菜单，设计效果令人兴奋、令人满意。

## 10.3 综 合 实 例

以“中国出版协会”(http://www.pac.org.cn/)网站的导航菜单为例，详细讲解网站页面构建的过程，效果如图 10-19 所示。

中国出版协会网站中二级导航菜单是采用 DIV+CSS+jQuery 技术实现的，此处改用 DIV+CSS 技术来实现，在实现过程中将所有超链接的 href 属性设置为“#”，并对原网站进

行简化，只设计网站的头部、导航区域、新闻图像显示区域、底部版权区域等，省略了其他区域的信息的设计。

**1．网站页面 div 布局设计**

```
<div class="body-top">
    <div class="header">
        <div class="logo">
            <div id="nav_wrap" >
                <div id="nav"> nav  </div>
            </div>
        </div>
    </div>
</div>
<div class="changediv">changediv</div>
<div class="footer">footer</div>
```




图 10-19　中国出版协会首页二级导航菜单效果图

**2．导航菜单结构设计**

```
<ul class="clearfix">
    <li><a href="">首页</a>|</li>        <!-- 一级导航  -->
    <li><span class="v"><a href="#">协会概况</a>   <!-- 一级导航  -->
        <div>
            <a href="#">协会简介</a>    <!-- 二级导航  -->
            <a href="#">大事记</a>      <!-- 二级导航  -->
            <a href="#">协会章程</a>    <!-- 二级导航  -->
            <a href="#">协会领导</a>    <!-- 二级导航  -->
            <a href="#">组织机构</a>    <!-- 二级导航  -->
            <a href="#">历史沿革</a>    <!-- 二级导航  -->
        </div>
    </li>
    <li><a href="">新闻公告</a>|</li>   <!-- 一级导航  -->
    ...
<ul>
```

### 3. 网站页面代码设计

```
<!-- edu_10_3_1.hmtl -->
<!doctype html>
<html lang="en">
    <head>
        <meta charset="UTF-8">
        <meta name="keywords" content="中国出版协会">
        <meta name="description" content="中国出版协会">
        <link rel="stylesheet" href="pac.css" type="text/css">
        <title>中国出版协会简化网站</title>
    </head>
    <body>
        <div class="body-top">
            <div class="header">
                <div class="logo">
                <div id="nav_wrap" >
                    <div id="nav">
                        <ul class="clearfix">
                            <li><span class="v">
                            <a href="#" target="_blank">首页</a>
                            <span class="cut_line">|</span></span>
                            </li>
                            <li><span class="v">
                                <a href="#">协会概况</a></span>
                                <span class="cut_line">|</span>
                                <div class="kind_menu" style="left: 40px">
                                    <a href="#">协会简介</a><span>|</span>
                                    <a href="#">大事记</a><span>|</span>
                                    <a href="#">协会章程</a><span>|</span>
                                    <a href="#">协会领导</a><span>|</span>
                                    <a href="#">组织机构</a><span>|</span>
                                    <a href="#">历史沿革</a><span>|</span>
                                </div>
                            </li>
                            <li><span class="v">
                                <a href="#">新闻公告</a></span>
                                <span class="cut_line">|</span>
                                <div class="kind_menu" style="left:40px">
                                <a href="#">协会动态</a><span>|</span>
                                <a href="#">行业动态</a><span>|</span>
                                </div>
                            </li>
                            <li><span class="v">
                                <a href="#">领导讲话</a><span class="cut_line">|</span></span>
                            </li>
                            <li><span class="v">
                                <a href="#">政策法规</a></span><span class="cut_line">|</span>
                                <div class="kind_menu" style="left: 40px">
                                    <a href="#">政策发布</a><span>|</span>
                                    <a href="#">法律法规</a><span>|</span>
                                </div>
                            </li>
                            <li><span class="v">
                                <a href="#">工作简报</a></span><span class=
```

```
"cut_line">|</span>
            <div class="kind_menu" style="left: 40px">
            </div>
        </li>
        <li><span class="v">
             <a href="#">评奖表彰</a></span><span class="cut_line">|</span>
            <div class="kind_menu" style="left: 40px">
            </div>
        </li>
        <li><span class="v">
             <a href="#">会员服务</a></span><span class="cut_line">|</span>
            <div class="kind_menu" style="left: 40px">
                <a href="#">会员单位</a><span>|</span>
                <a href="#">会员动态</a><span>|</span>
                <a href="#">申请入会</a><span>|</span>
                <a href="#">会员变更</a><span>|</span>
                <a href="#">会员风采</a><span>|</span>
                <a href="#">行业维权</a><span>|</span>
                <a href="#">行业自律</a><span>|</span>
                <a href="#">建言献策</a><span>|</span>
            </div>
        </li>
        <li><span class="v">
             <a href="#">教育培训</a></span><span class="cut_line">|</span>
            <div class="kind_menu" style="left: 40px">
            </div>
        </li>
        <li><span class="v">
              <a href="#">外事</a></span><span class="cut_line">|</span>
            <div class="kind_menu" style="left: 40px">
            </div>
        </li>
        <li><span class="v">
             <a href="#">专家访谈</a></span><span class="cut_line">|</span>
            <div class="kind_menu" style="left: 40px">
            </div>
        </li>
        <li><span class="v">
          <a href="#">展会</a></span><span class="cut_line">|</span>
            <div class="kind_menu" style="left: 40px">
            </div>
        </li>
        <li><span class="v">
            <a href="#">行业研究</a></span>
            <div class="kind_menu" style="left: 40px">
                <a href="#">高端视点</a><span>|</span>
                <a href="#">理论专题</a><span>|</span>
                 <a href="#">调研报告</a><span>|</span>
                 <a href="#">在线调研</a><span>|</span>
             </div>
         </li>
```

```
104                     </ul>
105                 </div><!--nav-->
106             </div><!--nav_wrap-->
107         </div>
108     </div>
109     </div>
110     <div class="changediv">
111         <a href="#"><img src="v9/20150213112820232.jpg" width="960" height="198" alt="中国出版年会召开" /></a>
112     </div>
113     <div class="footer">
114         <div class="" style="padding-top:10px;margin-bottom:10px;" >
115             <a  href="#">关于我们</a> |
116             <a  href="#">网站地图</a> |
117             <a  href="#">版权声明</a>  |
118             <a  href="#">人才招聘</a>
119         </div>
120         <div>
121           <span>备案号：京ICP备05020570号</span><span>版权所有：中国出版协会</span>
122           <span>技术支持：<a href="#">北京中青文文化传媒有限公司</a></span>
123         </div>
124         <div>
125             <span>办公地址：北京市东城区美术馆东街22号</span>
126             <span>邮编：100010</span><span>电话：010-65246062</span>
127             <span>电子邮箱：cbanxie@163.com</span>
128         </div>
129     </body>
130 </html>
```

上述代码中二级子菜单采用类名为 kind_menu 的 div 作为容器，内插入若干超链接作为二级子菜单，如代码中第 25～32 行、第 64～73 行等之间的 div 就是二级子菜单。页面装载时不显示子菜单，当鼠标滑过一级菜单中的列表项时，通过样式定义显示二级子菜单。IE6 以下浏览器不支持，Chrome、FireFox、IE7 以上浏览器均支持。

**4．对象的显示与隐藏 CSS 规则设计**

在 CSS 的布局中实现特定对象显示与隐藏方法有两种，分别介绍如下。

1）display 显示属性

设置或检索对象是否显示以及如何显示。

（1）基本语法。

```
display:block|none|inline
```

（2）语法说明。

block：用该值为对象之后添加新行。

none：与 visibility 属性的 hidden 值不同，其不为被隐藏的对象保留其物理空间。

inline：用该值将从对象中删除，以内联方式显示对象。

举例如下：

```
#div1{display:none;}  /* 让div1初始装载时不显示 */
#nav a:hover #div1{ display:block;} /* 鼠标滑过时div1显示 */
```

2）visibility 可视属性

设置或检索是否显示对象。与 display 属性不同，此属性为隐藏的对象保留其占据的物理空间。如果希望对象为可视，其父对象也必须是可视的。

（1）基本语法。

```
visibility : inherit|visible|collapse|hidden
```

（2）语法说明。

inherit：继承上一个父对象的可见性。

visible：对象可视。

hidden：对象隐藏。

collapse：主要用来隐藏表格的行或列。隐藏的行或列能够被其他内容使用。对于表格外的其他对象，其作用等同于 hidden。IE 5.5 尚不支持此属性。

举例如下：

```
img { visibility: hidden; float: right; }   /* 让对象隐藏*/
img { visibility: visible; float: right; }    /* 让对象恢复可视*/
```

**5．定义 pac.css 文件**

```
/* 网站：中国出版协会简化网站
   样式表文件名：pac.css
   应用对象：edu_10_3_1.hmtl
*/
body{ color: #010101; background: #fff;
  margin-top: 0px; margin-left: 0px; margin-right: 0px;}
body, html{ font: 12px/1.5 tahoma, arial, sans-serif;display: block;}
.body-top{height: 297px;background: url("v9/b1.jpg");}
.header,.header .logo{width: 960px;margin: 0 auto;
  height: 297px;background: url("v9/b2.jpg") no-repeat center;}
/*New Nav Style*/
#nav ul{margin:0px;padding:0px;}
#nav li{ text-align:center;font-size:14px;font-weight:700;}
#nav_wrap {width: 960px;overflow: hidden;padding-top: 223px;}
#nav{height: 69px;width: 960px; margin: 0 auto;padding:0px 5px;
         position: relative;}
#nav li {float: left;list-style: none;}
#nav li .v a{padding: 0 4px;height: 39px;line-height: 33px;
                  display: block;color: #0d2972;float: left;}
#nav li .v a:hover{color: #d62e38;text-decoration: none;}
#nav.kind_menu{height:30px;width:880px;top:26px;left:70px; line-height: 30px;
 vertical-align: middle; position: absolute; padding-top: 18px;  font-weight: normal; color: #152026; font-size: 12px;
 text-align: left; display:none; /* 初始不显示 */}
#nav .kind_menu a {color: #152026;text-align: center;
  padding: 0 10px; font-family: Arial, Helvetica, sans-serif;}
#nav .kind_menu a:hover {text-decoration: none;}
#nav .kind_menu span {font-size: 10px; color: #cecece; line-height: 30px;}
.cut_line{padding-top: 4px;display: inline-block;font-size: 14px;}
/* 鼠标滑过时显示二级菜单  */
#nav ul li:hover .kind_menu{display:block;left:100px; /* 显示子菜单  */}
```

```
31 .changediv{text-align:center;}
32 .footer{text-align:center;}
33 img{border:0px;}
```

上述代码中的第 21～23 行定义二级子菜单的显示样式。第 30 行定义当鼠标滑过时子菜单的样式，以块方式显示子菜单。

# 本 章 小 结

本章主要分析了常见的网站页面布局模式，给出每类模式的 DIV 结构设计和 CSS 文件编写方法。通过图层 div 合理地嵌套帮助初学者建立页面布局的概念，掌握常用页面布局结构编程方法。学会运用 CSS 样式文件来定义特定对象的样式，使所设计的网站页面能够尽量美观、漂亮，提升用户的体验。在进行样式定义时，最好能够学会使用浏览器兼容性测试工具来检查自己所编写的 CSS 规则，实现在不同浏览器上显示相同的页面效果。

# 练习与实验

## 练习 10

**1．选择题**

（1）下列 CSS 规则中能够让图层 div 不显示的选项是（　　）。

A．div{display:block;}　　B．div{display:none;}
C．div{display:inline;}　　D．div{display:hidden;}

（2）下列 CSS 规则中能够让列表项水平排列的选项是（　　）。

A．li{float:left;}　　B．li{float:none;}
C．li{float:middle;}　　D．li{float:up;}

（3）下列 CSS 规则中能够让图层 div 隐藏的选项是（　　）。

A．div{visibility:none;}　　B．div{ visibility: visible;}
C．div{visibility:hidden;}　　D．div{visibility:block;}

（4）下列 CSS 规则中能够使超链接在盘旋时产生上画线效果的选项是（　　）。

A．a:hover{text-decoration:none;}
B．a:hover{text-decoration:underline;}
C．a:hover{text-decoration:line-through;}
D．a:hover{text-decoration:overline;}

（5）下列 CSS 规则中能够使超链接在盘旋时下边框为 2px、实线、红色效果的选项是（　　）。

A．a:hover{border-bottom:2px solid #FF0000;text-decoration:none;}
B．a:hover{border-top:2px solid #FF0000;text-decoration:none;}
C．a:hover{border-bottom:2px dashed #FF0000;text-decoration:none;}
D．a:hover{border-right:2px double #FF0000;text-decoration:none;}

**2．简答题**

（1）简述采用 DIV+CSS 技术进行页面布局的基本步骤。

（2）说明 CSS 布局属性中“display:block”与“visibility：visible”的区别。

## 实验 10

1. 运用 DIV+CSS 技术实现如图 10-20 所示的页面布局。分别编写相应的 exp_10_1.html 和 CSS 外部样式表文件 exp_10_1.css。




图 10-20　Web 页面设计实例图

2．运用 DIV+CSS 完成如图 10-21 所示的页面布局。




图 10-21　DIV+CSS 布局实例

# 第 11 章　表　格

**本章学习目标**

使用文字、段落、图像和列表等元素进行网页设计，已经能够设计出一个基本的网页，但页面的信息元素不够丰富，特别是有些关联数据、同类数据等需要集中呈现时，仅仅使用列表、段落等标记不能很好地满足页面设计的需要，而表格是网页设计中常用的一种用于组织和排版同类或相关数据等信息的最好的集中呈现方式。通过表格可以精确地控制页面元素在网页中的位置。所以掌握表格标记及属性设置方法就显得十分重要。

Web 前端开发工程师应掌握以下内容:

- 掌握表格的标记和标记属性。
- 掌握表格行标记的属性及设置方法。
- 掌握表格单元格的跨行与跨列属性的设置方法，实现单元格合并。
- 学会使用表格和表格嵌套方法设计简易网页。

## 11.1　表格概述

在 Web 网页上如何将大量相关数据或同类数据组织起来并呈现给网络访问者呢？在 HTML 中可以使用表格 table 标记将一组相关数据直观、明了地展现给网络访问者。表格以简洁明了和高效快捷的方式将图片、文本、数据和表单的元素有序地显示在页面上，从而可以设计出漂亮的页面。

表格在网页设计中能将网页分成多个任意的矩形区域。定义一个表格时，使用成对<table></table>就可以完成，网页设计人员可以将任何网页元素放进 HTML 表格的单元格中。定义表格所使用的标记如表 11-1 所示。

**表 11-1　常用表格标记及说明表**

| 标　记 | 说　明 | 标　记 | 说　明 |
|---|---|---|---|
| <table></table> | 表格标记 | <thead></thead> | 定义表格的表头 |
| <caption></caption> | 表格标题标记 | <tbody></tbody> | 定义表格的主体 |
| <th></th> | 表格表头标记 | <tfoot></tfoot> | 定义表格的页脚 |
| <tr></tr> | 表格的行标记 | | |
| <td></td> | 表格的列标记 | | |

表格由表头、表体、表尾三部分组成。表头由若干个表格标题组成，表体由若干行组成，表尾由文字、相关数据和日期组成，标明表的设计单位、设计人和日期等信息。

**【例 11-1-1】**简易学生信息表。代码如下所示，页面效果如图 11-1 所示。

```
<!-- edu_11_1_1.html -->
<!doctype html>
<html lang="en">
    <head>
        <meta charset="UTF-8">
        <title>表格的定义</title>
    </head>
    <body>
        <table border="1" width="300px" height="100px">
            <tr>
                <th>姓名</th>
                <th>单位</th>
                <th>学号</th>
            </tr>
            <tr>
                <td>王小品</td>
                <td>商学院</td>
                <td>110204</td>
            </tr>
            <tr>
                <td>李白</td>
                <td>机械学院</td>
                <td>100244</td>
            </tr>
        </table>
    </body>
</html>
```


视频讲解

| 姓名 | 单位 | 学号 |
|---|---|---|
| 王小品 | 商学院 | 110204 |
| 李白 | 机械学院 | 100244 |

图 11-1　表格的定义

## 11.2　表 格 标 记

在 HTML 中，表格主要由五个标记构成：table、caption、tr、th、td 标记。

**1．基本语法**

```
<table>
    <caption>表格标题 </caption>
    <tr>
        <th></th>
        <th></th>
        <th></th>
    </tr>
    <tr>
        <td></td>
        <td></td>
        <td></td>
    </tr>
    ...
```

```
</table>
```

**2. 语法说明**

- table 标记是成对标记，<table>表示表格开始，</table>表示表格结束。
- caption 标记是成对标记，<caption>表示标题开始，</caption>表示标题结束。使用 caption 标记可以给表格添加标题，该标题应位于 table 标记与 tr 标记之间的任何位置。
- tr(Table Row)标记是成对标记，<tr>表示行开始，</tr>表示行结束。
- th(Table Heading 表头)标记是成对标记，<th>表示表头开始，</th>表示表头结束。设计表格时，表头常常作为表格的第 1 行或者第 1 列，用来对表格单元格的内容进行说明。表头文字内容一般居中、加粗显示。
- td(Table Data)标记是成对标记，定义单元格或列。以<td>开始，以</td>结束。表头可以用 th 标记定义，也可以用 td 标记定义，但<td></td>两标记之间的内容不自动居中、加粗。

在一个表格中，可以插入多个 tr 标记，表示多行，一组<tr>…</tr>标记表示插入一行。一行中可以有多个列，列（也称为单元格）中的内容可以是文字、数据、图像、超链接、表单元素等。

**【例 11-2-1】**设计班级课程表。代码如下所示，页面效果如图 11-2 所示。



视频讲解

```
<!--edu_11_2_1.html-->
<!doctype html>
<html lang="en">
   <head>
      <meta charset="UTF-8">
      <title>定义表格</title>
      <style type="text/css">
         td{text-align:center;}
         #bg{background:#E0E0E0;}
      </style>
   </head>
   <body>
      <table width="700" height="150px" border="1">
         <comment>表格标题</comment>
         <caption><strong>2012软件工程班课程表</strong></caption>
         <tr>
            <th>节次</th>
            <th>星期一</th>
            <th>星期二</th>
            <th>星期三</th>
            <th>星期四</th>
            <th>星期五</th>
         </tr>
         <tr id="bg">
            <td>第1-2节</td>
            <td>Java程序设计</td>
            <td>Web前端开发技术</td>
            <td>数字逻辑电路</td>
            <td>数据结构</td>
            <td>体育</td>
```

```
31            </tr>
32            <tr>
33                <td>第3-4节</td>
34                <td>心理咨询</td>
35                <td>线性代数</td>
36                <td>数据结构</td>
37                <td>数据结构</td>
38                <td>Web前端开发技术</td>
39            </tr>
40        </table>
41    </body>
42 </html>
```

**3．代码解释**

代码中第 13～40 行插入一个 3 行 6 列表格，其中第 15 行定义表格的标题；第 16～23 行定义表头，表头的内容居中加粗显示；第 24～31 行定义表格的第 2 行；第 32～39 行定义表格的第 3 行。其中表格第 2 行应用#bg 样式，加上背景效果。



2012软件工程班课程表

| 节次 | 星期一 | 星期二 | 星期三 | 星期四 | 星期五 |
|---|---|---|---|---|---|
| 第1-2节 | Java程序设计 | Web前端开发技术 | 数字逻辑电路 | 数据结构 | 体育 |
| 第3-4节 | 心理咨询 | 线性代数 | 数据结构 | 数据结构 | Web前端开发技术 |

图 11-2　插入表格

# 11.3　表格属性设置

表格是一种常用的页面布局方法，也是网页中数据分析的最好展示工具之一。实际应用中借助于表格标记和标记属性可以完成表格的装饰和美化。表格标记的属性如表 11-2 所示。

**表 11-2　表格标记的属性、取值及说明表**

| 属　　性 | 值 | 描　　述 |
|---|---|---|
| align | left\|center\|right | 规定表格相对周围元素的对齐方式 |
| bgcolor | #rrggbb\| colorname<br>rgb(r%,g%,b%)\| rgb(rr,gg,bb) | 规定表格的背景颜色 |
| border | pixels | 规定表格边框的宽度 |
| cellpadding | pixels\|% | 规定单元边沿与其内容之间的空白 |
| cellspacing | pixels\|% | 规定单元格之间的空白 |
| frame | above\|below\|hsides\| vsides\|<br>lhs\|rhs\| border\| void | 规定外侧边框的哪个部分是可见的 |
| rules | none\|all \|rows\|cols\|groups | 规定内侧边框的哪个部分是可见的 |
| height | %\|pixels | 规定表格的高度 |
| width | %\|pixels | 规定表格的宽度 |

### 11.3.1 表格边框属性

设置表格的边框属性可以改变表格的外观。边框属性如表 11-3 所示。表格中的属性同样适用于单元格。

表 11-3 表格边框属性说明表

| 边 框 属 性 | 说　　明 | 边 框 属 性 | 说　　明 |
| --- | --- | --- | --- |
| border | 表示表格边框粗细 | bordercolorlight | 表示表格亮边框颜色 |
| bordercolor | 表示表格边框颜色 | bordercolordark | 表示表格暗边框颜色 |

**1. 基本语法**

```
<table border="" bordercolor="" bordercolorlight="" bordercolordark="">…</table>
```

**2. 语法说明**

- border 属性：用于设置边框的粗细，单位是像素。
- bordercolor 属性：设置表格边框的颜色，可以使用 rgb 函数、十六进制数和颜色英文名称。
- bordercolorlight 属性：设置表格亮边框，对表格左上边框生效。
- bordercolordark 属性：设置表格暗边框，对表格右下边框生效。

### 11.3.2 表格的宽度和高度属性

通过 width 属性和 height 属性可以设置表格的宽度和高度。

**1. 基本语法**

```
<table width="" height=""> … </table>
```

**2. 语法说明**

- width：单位可以是长度单位或百分比，用于定义表格的宽度。
- height：单位可以是长度单位或百分比，用于定义表格的高度。
- 设置表格的高度与宽度为百分比时，表格会跟随浏览器窗口的改变而自动调整。

### 11.3.3 表格背景颜色与背景图像属性

设置表格的 bgcolor 属性可以改变表格的背景颜色，设置表格的 background 属性可以为表格增添背景图像效果，使表格更加美观。

**1. 基本语法**

```
<table bgcolor=" " background="" …> … </table>
```

**2. 语法说明**

- bgcolor：可以用 rgb 函数、十六进制、英文颜色名称来设置背景颜色。
- background：设置背景图像，图像的路径可以是绝对路径或相对路径。
- 同时设置背景颜色和背景图像属性时，背景图像会部分或完全覆盖背景颜色。

**【例 11-3-1】**设置表格边框属性。代码如下所示，页面效果如图 11-3 所示。




图 11-3　设置表格的边框属性

视频讲解

```
<!-- edu_11_3_1.html -->
<!doctype html>
<html lang="en">
 <head>
  <meta charset="UTF-8">
        <title>设置表格边框、背景、范围</title>
        <style type="text/css">
            h4{text-align:center;color:#0033cc;}
            td{text-align:center;}
        </style>
    </head>
    <body>
        <h4>设置表格边框、背景、范围</h4>
        <table align="center" width="700px" height="150px"
               border="15" bordercolor="#0000FF" bordercolorlight= "#ff0000"
bordercolordark="#6600ff" bgcolor="#99cccc" >
            <tr>
                <th>学号</th>
                <th>姓名</th>
                <th>所在院系</th>
            </tr>
            <tr>
                <td>110204</td>
                <td>王小品</td>
                <td>商学院</td>
            </tr>
            <tr>
                <td>100244</td>
                <td>李白</td>
                <td>机械学院</td>
            </tr>
        </table>
        <hr>
        <table align="center" border="15px"  width="700px" height="150px"
               background="backimage1.jpg"  bgcolor="#99cccc" >
            <tr>
                <th>学号</th>
                <th>姓名</th>
                <th>所在院系</th>
            </tr>
            <tr>
                <td>110204</td>
```

```
42                <td>王小品</td>
43                <td>商学院</td>
44            </tr>
45            <tr>
46                <td>100244</td>
47                <td>李白</td>
48                <td>机械学院</td>
49            </tr>
50        </table>
51    </body>
52 </html>
```

**3．代码解释**

代码中第 14～31 行、第 33～50 行分别定义两个 3 行 3 列的表。其中第 14 行定义表格的对齐方式、宽度和高度；第 15 行定义边框粗细、边框颜色、亮边框颜色、暗边框颜色、背景颜色；第 34 行同时设置了表格的背景颜色与背景图像，但背景图像覆盖了背景颜色。

## 11.3.4 表格边框样式属性

在表格中设置 table 标记中的 frame 属性可以改变表格边框的样式，设置 rules 属性可以改变表格内部边框的样式。

**1．基本语法**

```
<table frame=" " rules="" …> … </table>
```

**2．语法说明**

frame、rules 属性值及说明如表 11-4 所示。

**表 11-4　frame、rules 常见属性值及说明**

| frame 属性值 | 说　明 | rules 属性值 | 说　明 |
|---|---|---|---|
| above | 显示上边框 | all | 显示所有内部边框 |
| below | 显示下边框 | none | 不显示内部边框 |
| hsides | 显示上下边框 | rows | 仅显示行边框 |
| vsides | 显示左右边框 | cols | 仅显示列边框 |
| lhs | 显示左边框 | groups | 显示介于行列间边框 |
| rhs | 显示右边框 | | |
| border | 显示上下左右边框 | | |
| void | 不显示边框 | | |

**【例 11-3-2】**设置表格的边框样式属性。代码如下所示，页面效果如图 11-4 所示。

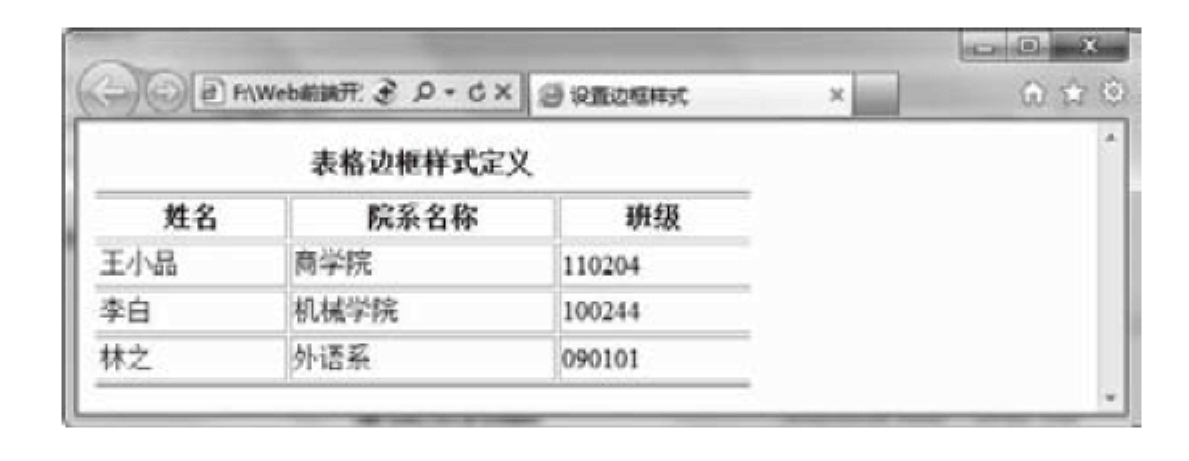



图 11-4　设置边框样式

```
1 <!-- edu_11_3_2.html -->
2 <!doctype html>
3 <html lang="en">
4     <head>
5         <meta charset="UTF-8">
6         <title>设置边框样式</title>
7     </head>
8     <body>
9         <table border="2" bordercolor="#00cccc" width="400px" height= "120px"
frame="hsides" rules="all">
10            <caption><b>表格边框样式定义</b></caption>
11            <tr>
12                <th>姓名 </th>
13                <th>院系名称 </th>
14                <th>班级</th>
15            </tr>
16            <tr>
17                <td>王小品 </td>
18                <td>商学院 </td>
19                <td>110204</td>
20            </tr>
21            <tr>
22                <td>李白 </td>
23                <td>机械学院 </td>
24                <td>100244</td>
25            </tr>
26            <tr>
27                <td>林之 </td>
28                <td>外语系 </td>
29                <td>090101</td>
30            </tr>
31        </table>
32    </body>
33 </html>
```

视频讲解

**3．代码解释**

代码中第 9 行设置表格的边框样式属性，设置 frame 属性值为 hsides，只显示表格上下边框，不显示左右边框。设置 rules 属性值为 all，显示表格内部的所有边框。

## 11.3.5　表格单元格间距、单元格边距属性

设置表格的 cellspacing 属性可以改变表格单元格之间的间隔，使网页中的表格内容稍微松散一些。设置表格的 cellpadding 属性可以增加表格的单元格的内容与内部边框之间的距离。

**1．基本语法**

```
<table cellspacing=""  cellpadding="" >  … </table>
```

**2．语法说明**

- cellspacing：值的单位为像素或百分比，默认值为 2px。
- cellpadding：值的单位为像素或百分比。

【例 11-3-3】设置表格的单元格间距和边距。代码如下所示，页面效果如图 11-5 所示。

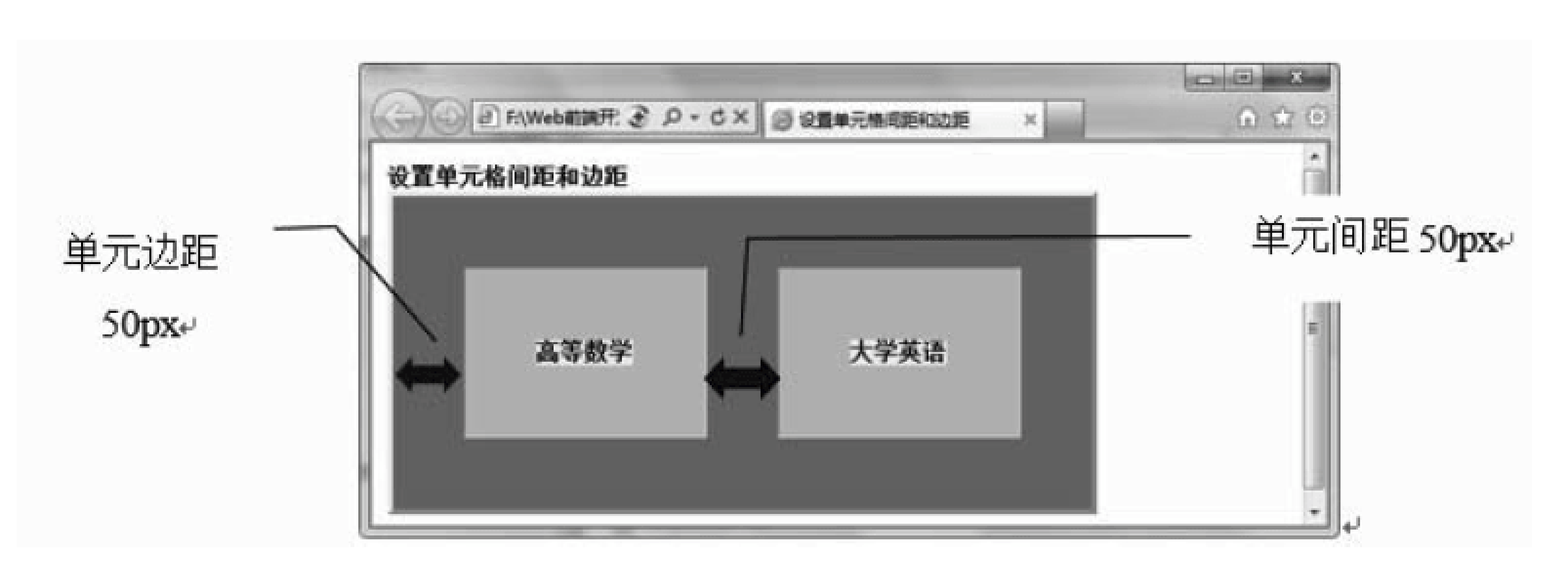



图 11-5　设置单元格间距

```
1 <!-- edu_11_3_3.html -->
2 <!doctype html>
3 <html lang="en">
4   <head>
5     <meta charset="UTF-8">
6     <title>设置单元格间距和边距</title>
7     <style type="text/css">
8       strong{background:#ccffcc;}
9       td{background:#99ccff;}
10    </style>
11  </head>
12  <body>
13    <b>设置单元格间距和边距</b>
14    <table width="500" border="4" cellspacing="50px" cellpadding="50px"
bgcolor="#9966ff">
15      <tr>
16        <td><strong>高等数学</strong></td>
17        <td><strong>大学英语</strong></td>
18      </tr>
19    </table>
20  </body>
21 </html>
```

视频讲解

**3．代码解释**

代码中第 8 行定义 strong 标记样式，作用是设置单元内容的背景颜色；第 9 行定义单元格的背景颜色；第 14 行设置了单元格的间距 50px、单元格边距为 50px。

## 11.3.6　表格水平对齐属性

通过表格标记的 align 属性可以设定表格在水平方向上的对齐方式，分别有居左、居中、居右三种。

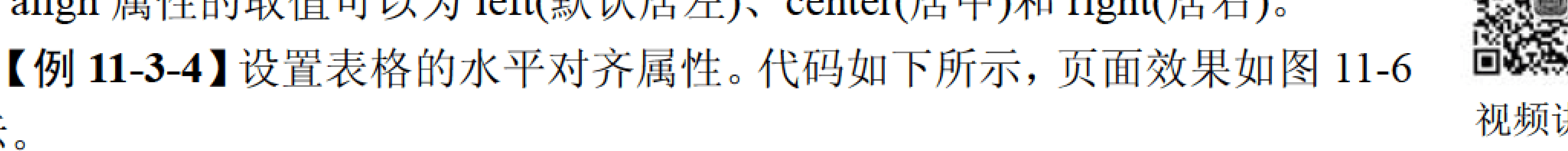
**1．基本语法**

```
<table align="left|center|right"> …</table>
```

**2．语法说明**

align 属性的取值可以为 left(默认居左)、center(居中)和 right(居右)。


视频讲解



【例 11-3-4】设置表格的水平对齐属性。代码如下所示，页面效果如图 11-6 所示。

```
<!-- edu_11_3_4.html -->
<!doctype html>
<html lang="en">
```

```
4      <head>
5          <meta charset="UTF-8">
6          <title>设置表格水平对齐方式</title>
7          <style type="text/css">
8              div{width:100%;height:100px;}
9          </style>
10     </head>
11     <body>
12         <div id="" class="">
13             <table align="left"  border="2">
14                 <caption>学生信息表(左对齐)</caption>
15                 <tr>
16                     <td>王小品 </td>
17                     <td>商学院 </td>
18                     <td>110204</td>
19                 </tr>
20                 <tr>
21                     <td>李白 </td>
22                     <td>机械学院 </td>
23                     <td>100244</td>
24                 </tr>
25             </table>
26         </div>
27         <div id="" class="">
28             <table  align="center"  border="2">
29                 <caption>学生信息表(居中对齐)</caption>
30                 <tr>
31                     <td>王小品 </td>
32                     <td>商学院 </td>
33                     <td>110204</td>
34                 </tr>
35                 <tr>
36                     <td>李白 </td>
37                     <td>机械学院 </td>
38                     <td>100244</td>
39                 </tr>
40             </table>
41         </div>
42     </body>
43 </html>
```




图 11-6　设置表格对齐方式

### 3．代码解释

代码中通过两个图层 div 设置两种表格水平对齐方式。第 12～26 行 div 中设置了表格左对齐；第 27～41 行 div 中设置了表格居中对齐。

# 11.4　设置表格行的属性

表格行 tr 标记的属性用于设置表格某一行的样式，其属性设置如表 11-5 所示。

表 11-5　行 tr 标记的属性表

| 属　性　值 | 说　　明 | 属　　性 | 说　　明 |
|---|---|---|---|
| align | 行内容水平对齐 | bordercolor | 行的边框颜色 |
| valign | 行内容垂直对齐 | bordercolorlight | 行的亮边框颜色 |
| bgcolor | 行的背景颜色 | bordercolordark | 行的暗边框颜色 |

通过 tr 标记的 align 属性可以设置行内容的水平对齐方式。水平对齐方式有居左对齐、居中对齐和居右对齐。通过 tr 标记的 valign 属性可以设置行内容的垂直对齐方式。垂直对齐方式有顶部对齐、居中对齐和底部对齐。

**1．基本语法**

```
<table align="center" >
   <tr align="left | center | right" valign="top | middle | bottom">
      <td>... </td>
      ...
   </tr>
   ...
</table>
```

**2．语法说明**

left 表示设置行内容居左对齐；center 表示设置行内容居中对齐；right 表示设置行内容居右对齐。top 表示设置行内容顶部对齐；middle 表示设置行内容居中对齐；bottom 表示设置行内容底部对齐。其中行垂直居中对齐属性值与行水平居中对齐属性值不同。

**【例 11-4-1】**设置表格行内容对齐属性。代码如下所示，页面效果如图 11-7 所示。




图 11-7　设置行内容水平对齐

```
<!-- edu_11_4_1.html -->
<!doctype html>
<html lang="en">
   <head>
      <meta charset="UTF-8">
```


视频讲解

```
6          <title>设置行内容对齐方式</title>
7          <style type="text/css">
8              td{background:#ccffcc;}
9          </style>
10     </head>
11     <body>
12         <table border="1" width="450px" height="240px" align="center" bordercolor=
"#6600ff">
13             <caption><b>学生信息表(设置表行内容对齐方式)</b></caption>
14             <tr >
15                 <th>姓名 </th>
16                 <th>院系名称 </th>
17                 <th>班级</th>
18             </tr>
19             <tr  align="left" valign="top">
20                 <td>王小品 </td>
21                 <td>商学院 </td>
22                 <td>110204</td>
23             </tr>
24             <tr align="center" valign="middle" >
25                 <td>李白 </td>
26                 <td>机械学院 </td>
27                 <td>100244</td>
28             <tr align="right" valign="bottom">
29                 <td>林之 </td>
30                 <td>外语系 </td>
31                 <td>090101</td>
32             </tr>
33         </table>
34     </body>
35 </html>
```

**3．代码解释**

代码中第 8 行设置单元格 td 标记的背景颜色；第 12 行定义表格；第 19 行设置表格行内容对齐方式为水平居左、垂直居顶；第 24 行设置行内容对齐方式为水平、垂直均居中；第 28 行设置行内容对齐方式为水平居右、垂直居底。

# 11.5　设置单元格的属性

表格列标记 td 的属性可以设置表格单元格的显示风格。常用的属性如表 11-6 所示。单元格的颜色、边框和对齐属性与行 tr 标记一样。

**表 11-6　单元格 td 标记的属性表**

| 属　性　值 | 说　　明 | 属　　性 | 说　　明 |
|---|---|---|---|
| align | 单元格内容水平对齐 | bordercolorlight | 单元格的亮边框颜色 |
| valign | 单元格内容垂直对齐 | bordercolordark | 单元格的暗边框颜色 |
| bgcolor | 单元格的背景颜色 | rowspan | 单元格跨行 |
| background | 单元格背景图像 | colspan | 单元格跨列 |
| bordercolor | 单元格的边框颜色 | width | 单元格宽度 |
| | | height | 单元格高度 |

### 11.5.1 表格单元格跨行属性

使用单元格 td 标记的 rowspan 属性可以设置单元格跨行合并。

**1. 基本语法**

```
<td rowspan="行数">...</td>
```

**2. 语法说明**

rowspan 属性可以设置单元格跨行。通过 rowspan="n"（n 是正整数），可以设置某一单元格跨 n 行，当前行下的 n-1 行内的单元格数量都需要减少一个，即少定义一个 td 标记。

### 11.5.2 表格单元格跨列属性

使用单元格 td 标记的 colspan 属性可以设置单元格跨列合并。

**1. 基本语法**

```
<td colspan="列数">...</td>
```

**2. 语法说明**

colspan 属性可以设置单元格跨列。通过 colspan="n"（n 是正整数），可以设置某一单元格跨 n 列，当前行内的单元格数量需要减少 n-1 个，即删除 n-1 个 td 标记。

**【例 11-5-1】**设置表格单元格合并。代码如下所示，页面效果如图 11-8 所示。


视频讲解

```
<!-- edu_11_5_1.html -->
<!doctype html>
<html lang="en">
    <head>
        <meta charset="UTF-8">
        <title>设置单元格跨列、跨行属性</title>
    </head>
    <body>
        <h3 align="center">设置单元格跨列、跨行属性</h3>
        <table border="1" width="500px" align="center" bordercolor="#3366ff">
            <caption>云计算与物联网会议日程安排表</caption>
            <tr align="center">
                <td colspan="2">上午</td>
                <td colspan="2">下午</td>
            </tr>
            <tr >
                <td>8:00-10:00</td>
                <td>10:10-12:00 </td>
                <td>14:00-16:00</td>
                <td>16:10-18:00</td>
            </tr>
            <tr align="center">
                <td rowspan="2">领导讲话 </td>
                <td>大会主题报告</td>
                <td>分会专题报告</td>
                <td rowspan="2">总结报告</td>
            </tr>
            <tr align="center">
                <td>专家报告</td>
```

```
30                <td>分组讨论</td>
31            </tr>
32            <tr align="center">
33                <td colspan="4">全天参观考察无锡国家物联网中心</td>
34            </tr>
35        </table>
36    </body>
37 </html>
```




图 11-8　设置单元格合并

**3．代码解释**

代码中第 10 行设置表格宽度、高度、居中对齐方式、边框颜色等。第 13 行和第 14 行设置单元格跨 2 列合并；第 23 行和第 26 行设置单元格跨 2 行合并；第 33 行设置单元格跨 4 列合并。

# 11.6　表 格 嵌 套

表格嵌套是一种常用的页面布局方式。利用表格嵌套可以设计比较复杂且美观的页面效果。通常情况下，使用表格嵌套时，表格不宜过多使用，否则会降低网站访问速度。表格嵌套一般采用在单元格内嵌套表格。

**1．基本语法**

```
 1 <table>
 2     <tr>
 3         <td>      <!--   单元格内嵌套表格   -- >
 4             <table>
 5                 <tr>
 6                     <td>...</td>
 7                     ...
 8                 </tr>
 9                 <tr>
10                     <td>...</td>
11                     ...
12                 </tr>
13             </table>
14         </td>
15        <td>...</td>
16      ...
17     </tr>
18     ...
19 </table>
```

**2．语法说明**

第 4～13 行为在第 1 个表格的单元格内嵌套一个表格。

**【例 11-6-1】**设置嵌套表格。代码如下所示，页面效果如图 11-9 所示。

视频讲解

```
<!-- edu_11_6_1.html -->
<!doctype html>
<html lang="en">
    <head>
        <meta charset="UTF-8">
        <title>表格嵌套</title>
        <style type="text/css">
            ul{list-style-type:none;}
            li{width:80px;background:#00ccff;}
            p{text-indent:2em;font-size:16px;}
        </style>
    </head>
    <body>
        <h4 align="center">表格嵌套</h4>
        <table width="660px" border="1" align="center" bordercolor="#3333ff">
            <tr>
                <td width="170px"> </td>
                <td width="360px" rowspan="3"><p>地铁4号线横穿南京，从河西到仙林，起点龙江站。根据施工计划，龙江片区要掏一个深达20多米、长520米的地铁枢纽站。目前，车站已经完成70%的体量，剩余的30%，将是建设难度最大的部分。昨天，地铁施工方特意“打招呼”：为了安全，下月起将加大围挡范围，可能影响到部分商户经营及市民的出行，至少要3个月。据介绍，4号线计划明年底通车。</p></td>
                <td width="120">新闻链接</td>
            </tr>
            <tr>
                <td>
                    <table width="100%" border="1" bordercolor="#33ff99">
                    <tr>
                    <td>科技</td>
                    </tr>
                    <tr>
                    <td>财经</td>
                    </tr>
                    <tr>
                    <td>探索</td>
                    </tr>
                    </table>
                </td>
                <td rowspan="2">
                <ul>
                <li><a href="http://www.baidu.com">百度</a></li>
                <li><a href="http://www.163.com">网易</a></li>
                <li><a href="http://www.sina.com">新浪</a></li>
                <li><a href="http://www.sohu.com">搜狐</a></li>
                </ul>
                </td>
            </tr>
            <tr>
                <td> </td>
            </tr>
        </table>
    </body>
</html>
```

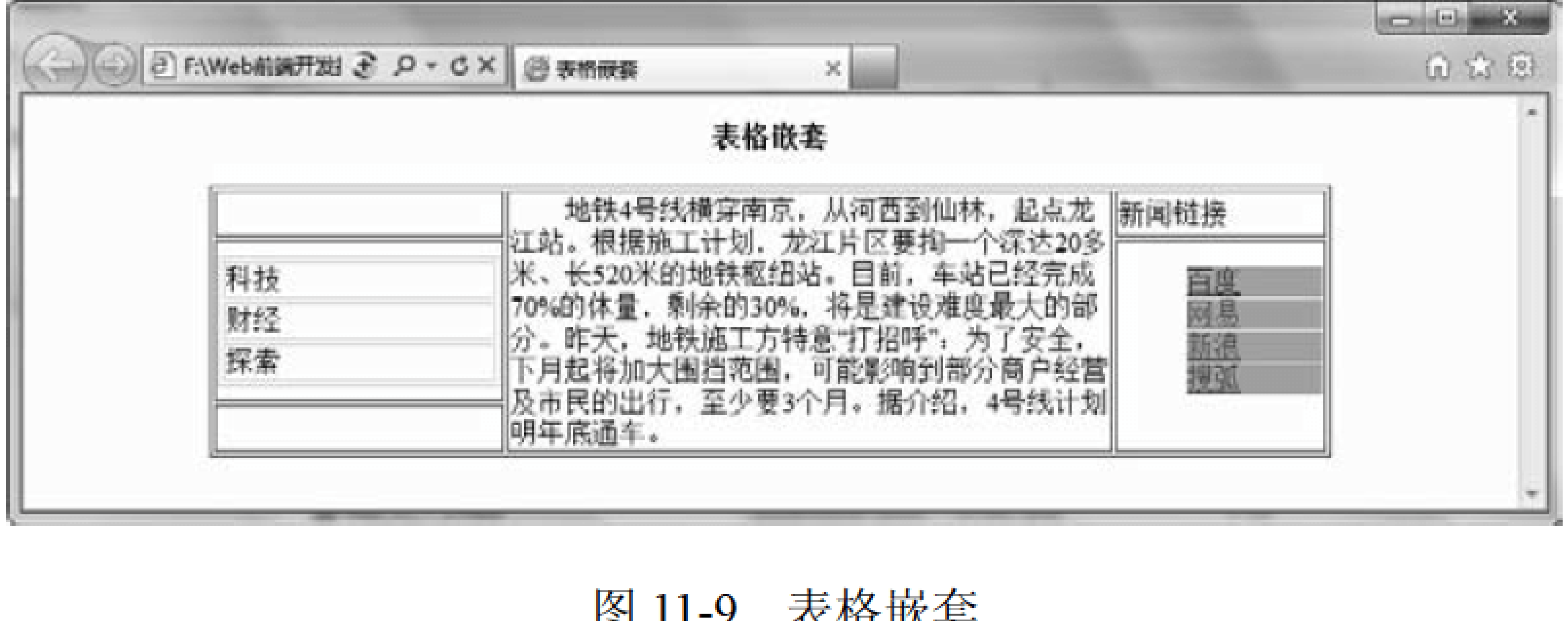



图 11-9　表格嵌套

### 3．代码解释

代码中第 15～47 行定义了一个 3 行 3 列的表格；第 23～33 行定义一个 3 行 1 列的表格，嵌套在第 1 个表的第 2 行第 1 列中。第 18 行设置单元格跨 3 行；第 35 行设置单元格跨 2 行。

## 11.7　综 合 实 例

以“医疗机械公司”网站为例，利用表格进行公司网站首页的布局设计，使用表格标记及标记属性的设置来美化表格，设计效果如图 11-10 所示。




图 11-10　医疗机械公司网站首页

### 1．首页页面表格布局

采用 5 行 2 列表格进行页面布局，在表格布局中使用单元格跨行、跨列合并以及单元

格嵌套表格等方法完成页面布局设计，其效果如图 11-11 所示。

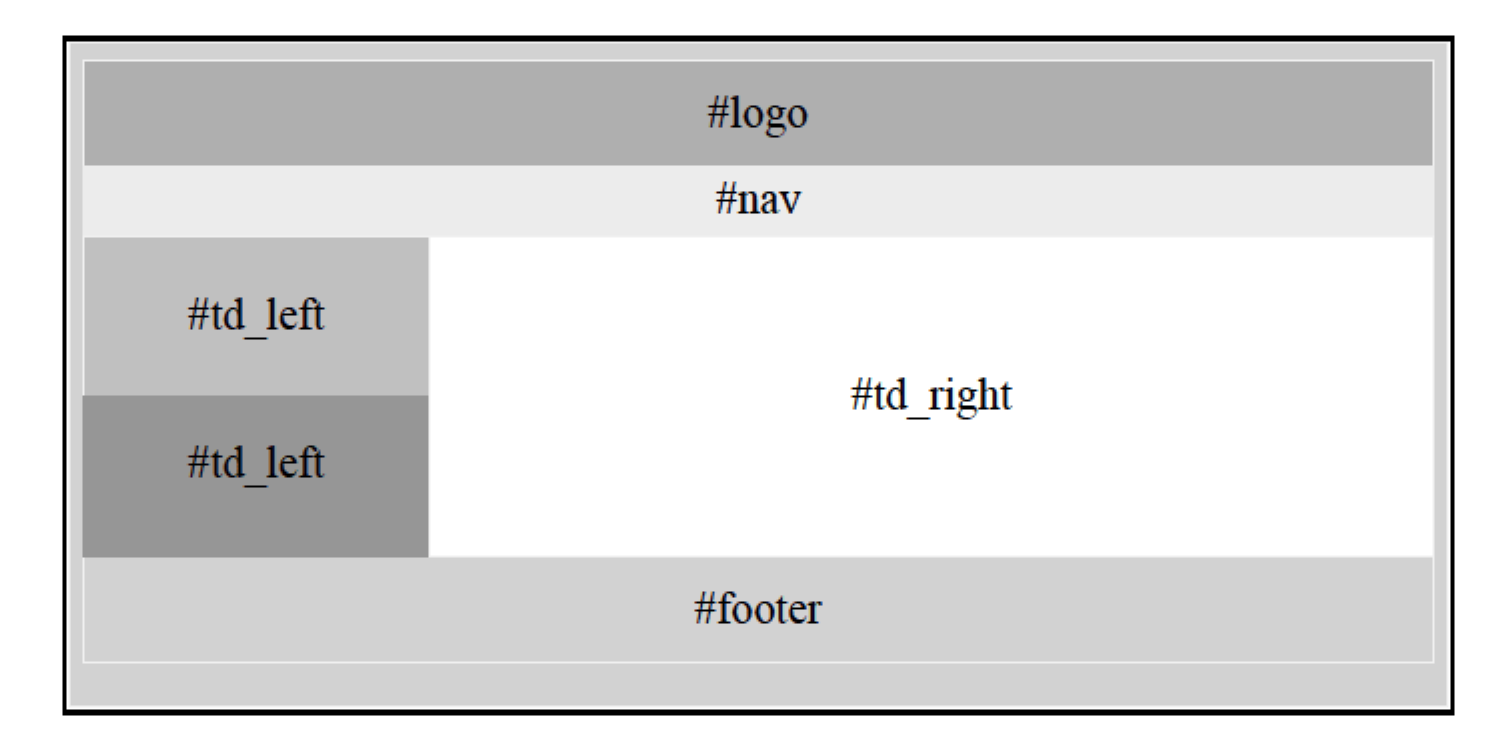



图 11-11　网站首页表格布局图

**2．网站首页 HTML 代码设计**

```
<!-- edu_11_7_2.html -->
<!doctype html>
<html lang="en">
   <head>
      <meta charset="UTF-8">
      <meta name="Description" content="医疗机械产品">
      <title>医疗机械公司网站</title>
      <link rel="stylesheet" href="edu_11_7_1.css" type="text/css">
   </head>
   <body>
      <table id="table1" align="center">
         <tr >
            <td colspan=2>
               <div id="logo">
                  <div id="floatl">
                     <img src="logo.png" alt="" border="0px"/>
                  </div>
                  <div id="floatr">
                     <a href='#' >设为首页</a>
                     <a href='#' >收藏本站</a>
                  </div>
               </div>
            </td>
         </tr>
         <tr id="nav">
            <td colspan="2">
               <table id="table2">
                  <tr>
                     <td><a href="#">首页</a></td>
                     <td><a href="#">关于我们</a></td>
                     <td><a href="#">新闻资讯</a></td>
                     <td><a href="#">产品展示</a></td>
                     <td><a href="#">招贤纳士</a></td>
                     <td><a href="#">联系我们</a></td>
                  </tr>
               </table>
            </td>
         </tr>
         <tr>
```

```
            <td colspan=2><img src="yiliao_big.jpg" alt=""></td>
        </tr>
        <tr>
        <td id="td_left">
            <dl>
                <dt><strong>关于我们</strong></dt>
                    <dd><a href="#">公司简介</a></dd>
                    <dd><a href="#">公司历史</a></dd>
                    <dd><a href="#">公司宗旨</a></dd>
                    <dd><a href="#">在线留言</a></dd>
                    <dd><a href="#">联系我们</a></dd>
            </dl>
        </td>
        <td rowspan="2" id="td_right">
            <p id="bt1"><strong>您现在的位置:</strong> 首页 > 关于我们 > 公司简介</p>
            <div id="div1">
                <a href="#" title="">
                <img src="qiyejianjie_small.jpg" style="width: 250px; height: 135px; margin: 10px; float: left;border:0px;" alt="" /></a><span style="font-size:12px;">医疗器械公司（前名为医疗器械厂），是一家专业研制、生产各类医用诊断X射线机和各种手术床的大中型国有法人独资企业。在国内医疗器械放射影像行业中具有很高的地位和影响。 本企业始建于1946年，具有60余年悠久历史，是国内大型医疗设备研发生产基地。历年来，公司先后研发生产的各类医院设备涉及医疗器械各个领域。目前，公司主导产品&mdash;&mdash;医用诊断X射线机的生产，属国内最早，产品范围覆盖5mA&mdash;800mA大、中、小型系列产品，每年各类X射线机 产量达2500台（套）。 </span>
            </div>
            <div> </div>
            <div id="div1">
                <span style="font-size:12px;">近年来，本企业抓住发展机遇，挖掘、发挥企业优势，加强企业内部各种要素的整合，优化资源配置，通过产、学、研的合作，通过和国外厂商经、技、贸的合作，使企业市场竞争能力不断增强。 目前，公司在全国30多个省市自治区设有营销、服务网点及代理机构70多个，覆盖全国及世界各个国家和地区，产品远销欧美、中东、东南亚等国家地区。同时，公司为国内外客户及部队用户提供开发、生产服务，为用户提供多种通用医疗配件和专用设备及满意的售后服务。 公司以专业的队伍、严谨的管理、卓越的设备、优质的服务，着力打造放射医疗设备界的品牌产品。
                </span>
            </div>
            <div>    </div>
            <div id="div1">
                <span style="font-size:12px;">在60多年的生产经营中，公司于1993年荣获高新技术企业的荣誉，并一直保持至今，荣获了首批科技小巨人企业，诚信企业，A类财务会计信用单位等荣誉。自1998年6月始，公司先后通过了ISO9001、ISO13485、CCC、 SFDA、CE等系列国际质量体系认证和产品认证，多项产品列入中国医疗装备协会推荐产品，《医用诊断X射线》机被推荐为上海名牌产品，本品牌被推荐为装备制造业与高新技术产业自主创新品牌，享有极高的声誉和良好的品牌效应。 医疗器械公司坚持用企业文化提升企业核心竞争力，使企业在发展中树立良好的社会形象。&ldquo;行远必自迩、追求无止境&rdquo;。公司将一如既往地励精图治，开拓创新，为医疗卫生事业，做出我们永远的承诺。 </span>
            </div>
            </td>
        </tr>
        <tr>
            <td id="td_left">
                <p id="bt1"><strong>联系方式</strong></p>
                <b>医疗器械公司</b>
```

```
74                  <ul>
75                      <li>电    话：021-654300(总机)</li>
76                      <li>传    真：021-654301(总经办)</li>
77                      <li>网    址：http://www.888.cn</li>
78                      <li>邮    箱：mbpeizhi@163.com</li>
79                      <li>地    址：上海市杨浦区蓝天路</li>
80                      <li>邮    编：200000</li>
81                      <li>营 销 部 ：021-65434(经理室)</li>
82                      <li>售后服务：021-65189</li>
83                  </ul>
84              </td>
85          </tr>
86          <tr>
87              <td colspan="2" >
88                  <div id="footer">
89                      <ul>
90                          <li>医疗器械 版权所有 2017-2020 湘ICP备8888888 </li>
91                              <li> 地址:上海市杨浦区蓝天路 电话:021-654300 E-mail:mbpeizhi@163.com</li>
92                              <li> Powered by MetInfo 5.2.10 ?2017-2020 www.metinfo.cn</li>
93                      </ul>
94                  </div>
95              </td>
96          </tr>
97      </table>
98  </body>
99 </html>
```

上述代码中第 11～97 行定义一个 5 行 2 列的表格作为网页的基本布局。第 25～38 行定义单元格跨列合并后嵌套一个 1 行 6 列的表格，用作导航菜单，省略了二级导航菜单的设计。第 44～51 行采用定义列表定义垂直导航菜单。整个页面设计中采用无序列表、定义列表、表格单元格合并、表格嵌套、图像嵌入、图层嵌套等技术综合实现。要使页面达到实际布局效果，还需要进行 CSS 样式定义。

**3．CSS 样式定义**

根据表格布局图，分别对表格的不同单元格进行样式定义，CSS 规则如下所示：

```
/* edu_11_7_1.css */
*{font-size:12px;padding:0px;margin:0px;}
a{font-size:14px;}
#table1{padding:0px;margin:0px auto;width:1004px;border:0px;}
#logo{background:#D3D6DD;width:1004px;height:40px;}
#floatl{float:left;width:141px;height:35px;}
#floatr{float:right;text-align:right;width:862px;height:25px;padding:5px 0px;}
#floatr a{text-decoration:none;color:#000000;}
#table2{padding:0px;margin:0px;width:100%;height:40px;text-align:center;}
#nav{background:url("bg_blue.jpg");text-align:center;}
#nav a:link,a:visited,a:hover,a:active{text-decoration:none;
       color:#FFFFFF;font-size:14px;font-weight:bold;}
#nav td{text-align:center;vertical-align:middle;width:15%;}
#td_left{width:24%;
height:145px;font-size:14px;background:#ECF0F1;color:#67F78;}
#td_left a:link,a:visited,a:hover,a:active{color:#000000;text- decoration:none;}
```

```
#td_right{padding:25px;width:76%;height:350px;font-size:12px;color:
#67F78;}
#td_left ul,dl{height:180px;line-height:1.5em;}
li,dd,b{padding-left:25px;display:block;color:#67F78;}
dt,#bt1{background:url("bt_bg_small.jpg") left center no-repeat;
            font-size:16px;margin-left:15px;padding:5px auto;height:
24px;
        border-bottom:1px solid #D2D2D2;margin:10px auto;}
#img_bg{width:5px;height:24px;padding:2px;border:0px;}
strong{padding-top:0px;font-size:16px;font-weight:bold;}
#div1{line-height:1.5em;text-indent:2em;color:#67F78;}
#footer{text-align:center;margin:15px auto;/* footer  */}
#footer ul li{list-style:none;line-height:1.5em;}
```

# 本 章 小 结

本章主要介绍了设计表格的所有标记和标记属性。

在进行表格设计，需要考虑好表格的对齐方式设计。表格的对齐方式分三类：表格 table 标记的 align 属性、行 tr 标记的 align 和 valign、列（单元格）td 标记的 align 和 valign。这些属性的设置如果使用 CSS 样式进行定义，效果更好。

设计表格的背景颜色与背景图像时，最好采用 CSS 样式表，这样效果更易控制。

由于表格的单元格内的内容不同，如果插入大的图像或视频文件时网络延迟会很大，易造成网页打不开，影响网站的正常访问。通常采用表格进行布局时，会使用表格嵌套来细化页面布局。表格嵌套时，必须在单元格中嵌入表格。

# 练习与实验

## 练习 11

**1．选择题**

（1）设置围绕表格的边框宽度的正确的标记是（　　）。

A．<table size="">　　B．<table border="">

C．<table bordersize="">　　D．<tableborder="">

（2）定义表头的标记是（　　）。

A．<table> </table>　　B．<td></td>

C．<tr></tr>　　D．<th></th>

（3）下列标记中能够实现跨多行的是（　　）。

A．<th colspan=""> </th>　　B．<tr rowspan=""></th>

C．<td colspan=""></td>　　D．<td rowspan=""></td>

（4）设置表格的背景图像正确的标记是（　　）。

A．<tr background="">　　B．<table background ="">

C．<th src="">　　D．<tr src="">

（5）能够设置表格的标题的标记是（　　）。

A．<tbody> </tbody>　　B．<tfoot></tfoot>

C．<thead> </thead>　　D．<caption></caption>

（6）设置表格行垂直居中的标记是（　　）。

A．<tr align="center" >　　B．<tr valign="middle" >

C．< tr align="middle" >　　D．<tr valign="center ">

**2．填空题**

（1）表格的标题标记是__________，表格行的标记是_______，单元格的表头标记是_______。

（2）单元格跨 3 行，设置格式为<td _________ ="_____"></td>；单元格跨 5 列，设置格式为<td _________ ="______"></td>。

（3）表格的外部边框样式可以使用________属性来定义，表格内部边框样式可以使用_______属性来定义。

（4）单元格边距属性是_________；单元格间距属性是_________。

**3．简答题**

（1）写出定义表格的所有常用标记，并说明各自的作用。

（2）写出定义表格边框的所有属性，并说明其作用。

（3）表格行对齐方式有几类？它们的属性取值有什么不同？

## 实验 11

1．编写 HTML 代码，实现如图 11-12 所示页面效果。要求使用 CSS 样式表统一定义 table 和 td 标记样式，分别如下：

- table 标记样式：边框为 8px、线型为双线、颜色为#0000ff。
- td 标记样式：边框为 1px、线型为 solid、颜色为 black、水平居中对齐。
- 两个嵌套表格背景颜色分别为#ffffee 和#fefefe。
- 外表宽度为 300px、居中对齐、单元间距和单元格边距均为 0。
- 两个子表宽度为 80%、居中对齐、边框为 1px。

2．采用表格布局完成 CASIO 计算器外观设计，其中表格的每一个单元格均需要设计带边框，效果如图 11-13 所示。

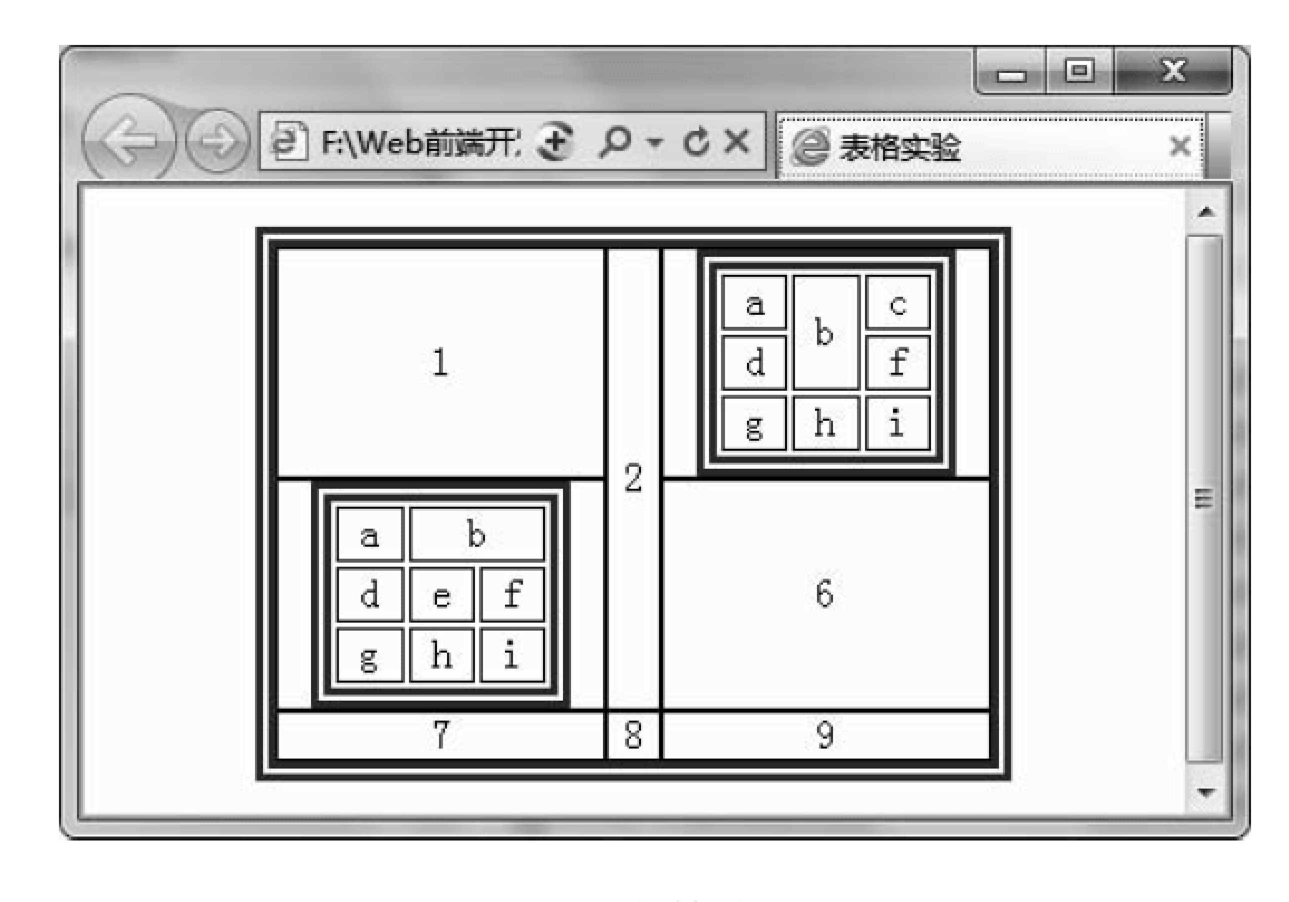



图 11-12　表格布局设计

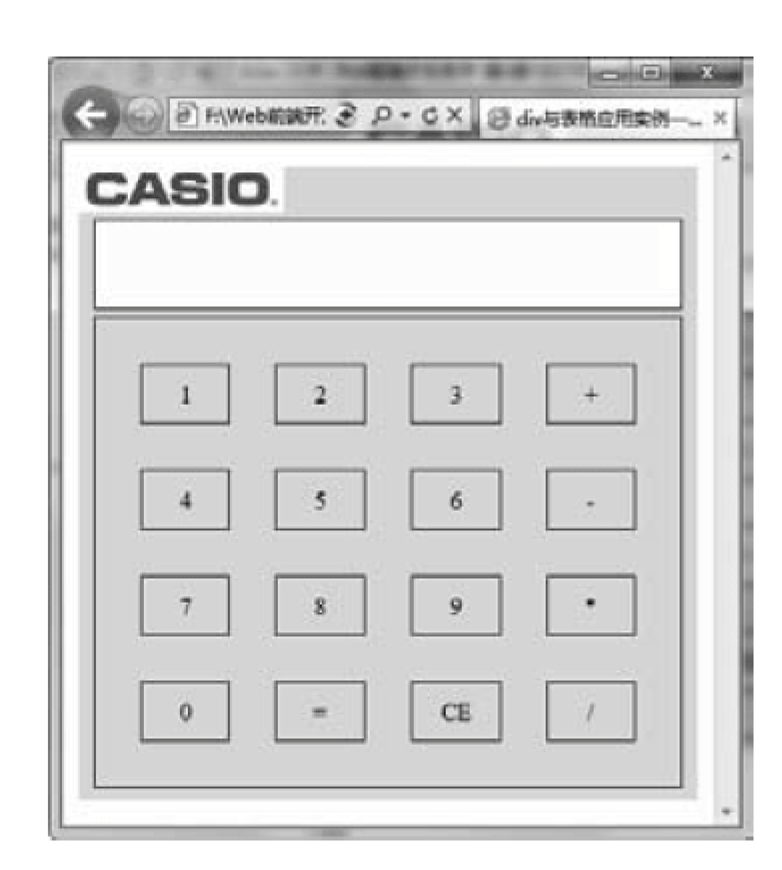



图 11-13　计算器页面布局设计

# 第 12 章　表　单

本章学习目标

运用 CSS+DIV 技术可以根据用户需求设计各式各样、丰富多彩的网站，用户通过浏览器去浏览网站的信息，但这样的网站仅是信息的发布者和提供者，用户也只是网站信息的浏览者，网站无法与用户进行交互。如果需要通过网站采集用户的有关信息或用户需要向网站管理员反馈相关信息，除了使用邮件之外，最有效的方法就是在网站上设计表单。表单可以让用户在线提交相关信息给服务器。服务器接收到信息之后，进行相应业务处理后再将处理结果返回给用户或管理者。

Web 前端开发工程师应掌握以下内容:

- 理解 Web 网页中表单的概念与作用。
- 掌握表单结构语法及属性语法。
- 掌握表单控件(元素)标记语法及属性语法。
- 掌握域和域标题标记语法。
- 学会综合运用表单及表单控件(元素)设计 Web 网页。

## 12.1　表单概述

Web 网页中的表单一般用来做网络调查、用户在线注册、信息检索及网站服务提供商向用户采集信息等。表单是较为复杂的 HTML 元素，经常与脚本、动态网页、后台数据处理等结合在一起使用，是设计动态网页的必备元素。利用表单可以在 HTML 页面中插入一些表单控件（元素），如文本框、提交按钮、重置按钮、单选按钮、复选框、下拉列表框等，完成各类信息的采集。

### 12.1.1　表单标记

表单 form 标记为成对标记，以<form>开始和</form>结束。表单定义了采集数据的范围，其所包含的数据内容将被完整地提交给服务器。

**1．基本语法**

```
<form method="post" action="">
   <input type="text" name="">
   <textarea name="" rows="" cols=""></textarea>
   <select name="">
      <option value="" selected></option>
      <option value=""></option>
   </select>
</form>
```

### 2．语法说明

<form>和</form>之间可包含各种表单信息输入标记。代码中第 2 行是单行文本输入框、第 3 行是多行文本域、第 4～7 行是下拉列表框。

### 3．属性说明

表单标记的属性主要有 name、action、method、enctype 等，其属性、取值及说明如表 12-1 所示。

**表 12-1　表单标记属性、取值及说明**

| 属　　性 | 值 | 说　　明 |
|---|---|---|
| name | name | 规定表单的名称 |
| action | url | 规定当提交表单时，向何处发送表单数据 |
| method | get\|post | 规定如何发送表单数据。post 方法主要包含名称/值对，并且无须包含于 action 属性的 URL 中。get 方法把名称/值对加在 action 的 URL 后面并且把新的 URL 送至服务器，不推荐使用 |
| enctype | MIME_type | 规定表单数据在发送到服务器之前应该如何编码 |

**【例 12-1-1】**表单的应用。代码如下所示，其页面效果如图 12-1 所示。

视频讲解

```
1 <!-- edu_12_1_1.html -->
2 <!doctype html>
3 <html lang="en">
4     <head>
5         <meta charset="UTF-8">
6         <title>表单的使用实例</title>
7     </head>
8     <body>
9         <form name="form1" method="post" action="form_action.jsp" enctype=
"text/plain">
10            <h3>输入课程成绩</h3>
11            姓名:<input type="text"/><br/>
12            高等数学:<input type="text" size="15"/>
13            大学物理:<input type="text" size="15"/><br/><br/>
14            <input type="submit" value="成绩提交"/>
15            <input type="reset" value="成绩重置"/>
16        </form>
17    </body>
18 </html>
```

### 4．代码解释

代码中第 9～16 行定义了一个表单，指定该表单的名称为 form1，提交方式为 post，处理程序为 form_action.jsp，编码方式为 text/plain；第 11～13 行定义了三个单行文本输入框，用于输入学生的姓名和课程成绩；第 14 行定义一个提交按钮；第 15 行定义一个重置按钮。

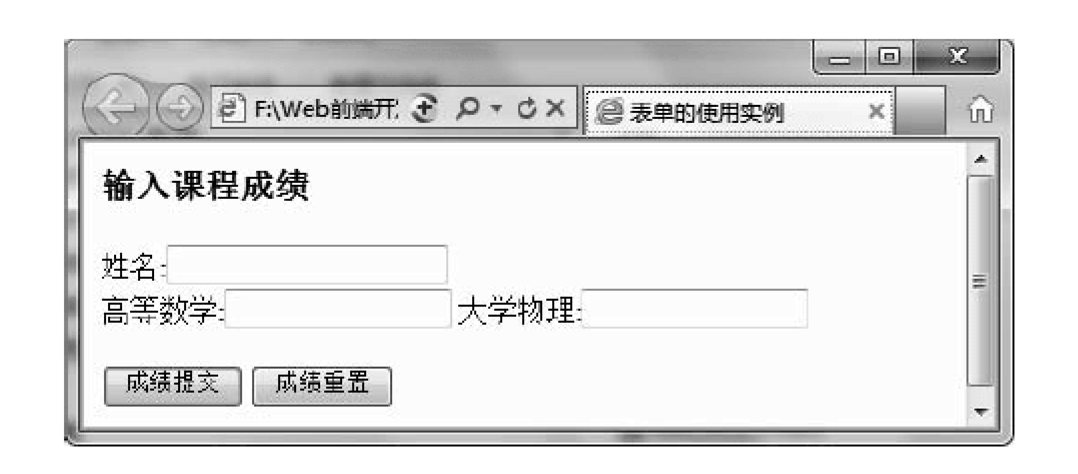



图 12-1　表单使用实例

## 12.2　定义域和域标题

利用 fieldset 标记可以在网页上定义域，在表单中使用域可以将表单的相关元素进行分组。fieldset 标记将表单内容的一部分打包，生成一组相关表单的字段。当一组表单元素放到 fieldset 标记内时，浏览器会以特殊方式来显示它们，它们可能有特殊的边界、3D 效果，或者可创建一个子表单来处理这些元素。legend 标记为 fieldset 标记定义域标题。

**1．基本语法**

```
<form>
   <fieldset>
      <legend align="left|center|right">域标题内容</legend>
   </fieldset>
</form>
```

**2．属性语法**

fieldset 标记没有属性，是成对标记。legend 标记必须位于 fieldset 标记内，也是成对标记；有一个对齐 align 属性，属性值分别为 left、center、right。

【例 12-2-1】域和域标题标记的应用。代码如下所示，页面效果如图 12-2 所示。



视频讲解

```
<!-- edu_12_2_1.html -->
<!doctype html>
<html lang="en">
   <head>
      <meta charset="UTF-8">
      <title>定义域和域标题实例</title>
   </head>
   <body>
      <form>
         <fieldset>
            <legend align="center">基本信息</legend>
            姓名：<input name="name" type="text">
            性别：<input name="sex" type="text">
         </fieldset>
         <fieldset>
            <legend align="center">其他信息</legend>
            身高：<input name="height" type="text">
            体重：<input name="weight"type="text">
         </fieldset>
      </form>
```

```
21      </body>
22  </html>
```




图 12-2　域和域标题的应用实例

**3．代码解释**

代码中第 10～14 行定义了一个域，域标题为“基本信息”，包含姓名和性别信息；第 15～19 行定义了另外一个域，域标题为“其他信息”，包含身高和体重信息。

# 12.3　表单信息输入

表单的主要功能是为用户提供输入信息的接口，将输入信息发送到服务器并等待服务器响应。表单中输入信息的标记是 input 标记，可以输入一行信息。input 标记是单个标记。

**1．基本语法**

```
<input name="" type="" />
```

**2．属性说明**

表单输入信息标记的属性主要有 name、type 等，输入类型是由类型 type 属性定义的。type 属性有很多不同的值，设置属性值不同，就会产生不同界面效果。input 标记的属性、取值及说明如表 12-2 所示。

**表 12-2　表单信息输入标记属性、取值及说明**

| 属　性 | 值 | 说　明 |
|---|---|---|
| name | name | 定义 input 元素的名称 |
| type | text \|password \|<br>checkbox \|radio \|<br>image \|submit \|reset \|<br>button \|file \|hidden | 规定 input 元素的类型。text:单行文本输入框；password:密码输入框；checkbox:复选框；radio:单选按钮；image:图像按钮；submit:提交按钮；reset:重置按钮；button:普通按钮；file:文件选择框；hidden:隐藏框 |

## 12.3.1　单行文本输入框

设置 input 标记的 type 属性值为 text，可以实现向表单中插入一个单行文本框。在单行文本框中可以输入任意类型的数据，但是输入的数据只能单行显示，不能换行。

**1．基本语法**

```
<input name="" type="text" maxlength="" size="" value="" readonly />
```

**2．属性说明**

单行文本输入框的主要属性有 name、maxlength、size、value、readonly，其属性、取

值及说明如表 12-3 所示。

**表 12-3　文本输入框标记属性、取值及说明**

| 属　　性 | 值 | 说　　明 |
| --- | --- | --- |
| name | name | 定义 input 元素的名称 |
| maxlength | number | 规定输入字段中的字符的最大长度 |
| size | number_of_char | 定义输入字段的宽度。其值小于或等于最大长度 |
| value | value | 规定 input 元素的默认值 |
| readonly | readonly | 规定文本框中内容只读，不能修改和编辑 |

## 12.3.2　密码输入框

设置 input 标记的 type 属性值为 password，可以实现向表单中插入一个密码输入框。密码输入框中可以输入任意类型的数据，与单行文本输入框有所不同，这些数据不是显式地显示在页面上，而是被显示字符“ • ”所取代，这样设计可以保障用户输入的密码不被泄露。

### 1．基本语法

```
<input name="" type="password" maxlength="" size=""/>
```

### 2．属性说明

密码输入框的主要属性、取值及说明如表 12-4 所示。

**表 12-4　密码输入框属性、取值及说明**

| 属　　性 | 值 | 说　　明 |
| --- | --- | --- |
| name | name | 定义 input 标记的名称 |
| maxlength | number | 规定输入字段中的字符的最大长度 |
| size | number_of_char | 定义输入字段的宽度。其值小于或等于最大长度 |

**【例 12-3-1】**用户信息输入。代码如下所示，页面效果如图 12-3 所示。

视频讲解

```
<!-- edu_12_3_1.html -->
<!doctype html>
<html lang="en">
    <head>
        <meta charset="UTF-8">
        <title>单行文本输入框实例</title>
    </head>
    <body>
        <h4>输入用户信息</h4>
        <form>
            用户名：<input type="text" name="chu" maxlength="20" size= "10"/>
            身份：<input type="text" name="" readonly value="学生"><br>
            密    码:<input type="password" name= "psw" maxlength="20" size="10">
        </form>
    </body>
</html>
```

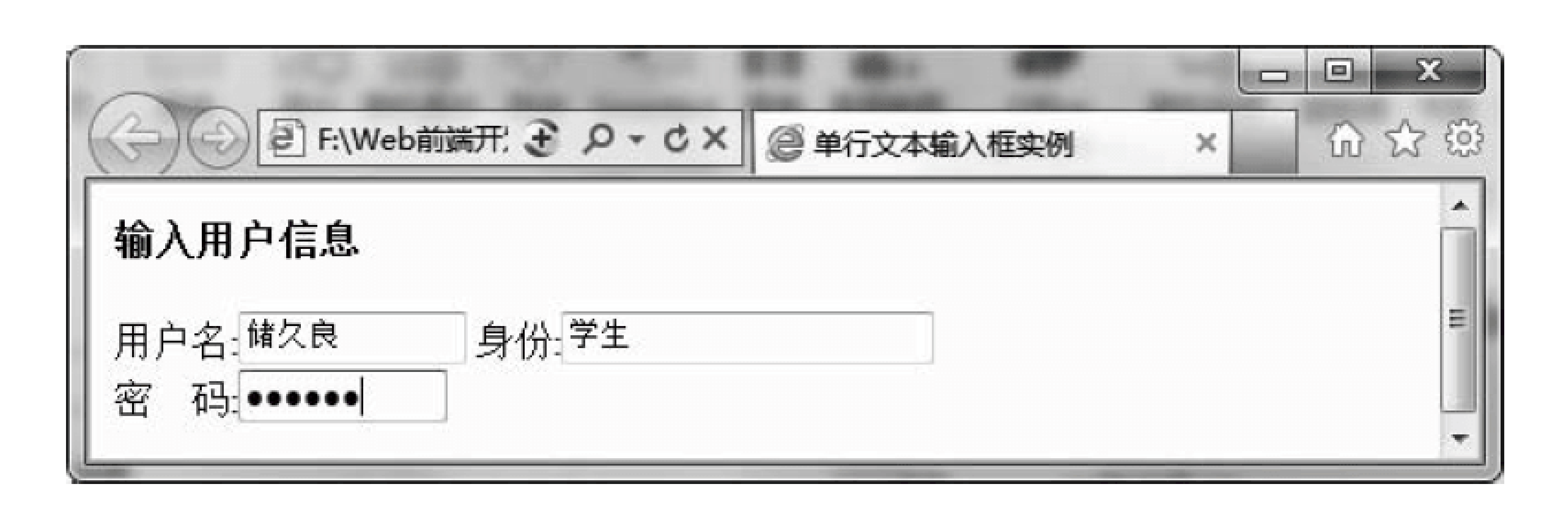



图 12-3　输入用户信息

**3．代码解释**

代码中第 11 行在表单中插入一个单行文本输入框，其名称为 chu，并定义最大长度为 20、显示宽度为 10，将超出宽度时，输入内容向左移动，直到达到最大长度为止，文本框的默认值为空；第 12 行插入一个单行文本框，赋初值为“学生”，且定义了 readonly 属性，此文本框不可修改。第 13 行插入一个密码框，其名称为 psw，并定义最大长度为 20、显示宽度为 10，将超出宽度时，输入内容向左移动，直到达到最大长度为止。密码输入框中输入的字符显示为“ • ”。

## 12.3.3　复选框

设置 input 标记的 type 属性值为 checkbox，可以实现向表单中插入一个复选框，用户可利用复选框在网页上设置多项选择。

**1．基本语法**

```
<input name="" type="checkbox" value="" checked="checked"/>
```

**2．属性说明**

复选框的主要属性有 name、value、checked，其中 checked 属性用于设置初始预选项。复选框的属性、取值及说明如表 12-5 所示。

**表 12-5　复选框属性、取值及说明**

| 属　　性 | 值 | 说　　明 |
|---|---|---|
| name | name | 定义 input 标记的名称 |
| value | value | 规定 input 标记的值 |
| checked | checked | 预先选定复选框 |

由于复选择框可以支持多选，每一个复选框都是不同的，一组复选框的所有 name 属性值应该不同，value 属性值也应该不同。

## 12.3.4　单选按钮

设置 input 标记的 type 属性值为 radio，可以实现向表单中插入一个单选按钮，用户可利用单选按钮在网页上为某一选择设置多个单选项。

**1．基本语法**

```
<input name="" type="radio" value="" checked="checked"/>
```

2．属性说明

单选按钮的属性有 name、value 和 checked 等，其属性、取值及说明如表 12-6 所示。

表 12-6　单选按钮属性、取值及说明

| 属　　性 | 值 | 说　　明 |
| --- | --- | --- |
| name | name | 定义 input 标记的名称 |
| value | value | 规定 input 标记的值 |
| checked | checked | 预先选定单选按钮 |

由于单选按钮必须是唯一的，所以在一组单选按钮中，只能选择一个单选按钮，所以一组单选按钮的所有 name 属性值必须相同，value 属性取值应该不同。

【例 12-3-2】复选框与单选按钮的应用。代码如下所示，页面效果如图 12-4 所示。

视频讲解

```
 1 <!-- edu_12_3_2.html -->
 2 <!doctype html>
 3 <html lang="en">
 4    <head>
 5        <meta charset="UTF-8">
 6        <title>复选框与单选按钮的应用</title>
 7        <style type="text/css">
 8        fieldset{width:300px;height:120px;border:2px double #003399;
padding-left:30px;}
 9        </style>
10    </head>
11    <body>
12        <form>
13            <fieldset>
14                <legend>请填写个人信息</legend><br>
15                姓名：<input type="text" name="xm" maxlength="10" size= "10"><br>
16                爱好：<input type="checkbox" name="c1" value="读书"/>读书
17           <input type="checkbox" name="c2" value="唱歌"checked= "checked"/>唱歌
18      <input type="checkbox" name="c3" value="游戏" checked="checked"/>游戏<br>
19       性别：<input type="radio" name="sex" value="male" checked="checked"/>男性
20                <input type="radio" name="sex" value="female"/>女性
21            </fieldset>
22        </form>
23    </body>
24 </html>
```

3．代码解释

代码中第 8 行定义 fieldset 标记的样式；第 13～21 行在表单中插入域和域标题标记，对表单元素进行分组，其中第 15 行在表单中插入单行文本输入框；第 16～18 行分别在表单中插入三个复选框，name 属性值分别为 c1、c2 和 c3，value 属性取值分别为“读书”“唱歌”和“游戏”，并给 input 标记设置 checked 属性，将名称为 c2 和 c3 复选框设置为预选

项；第 19～20 行在表单中插入两个单选按钮，name 属性值均为 sex，value 属性值分别为 male 和 female，并给 input 标记设置 checked 属性，将“男性”单选按钮设置成预选项。

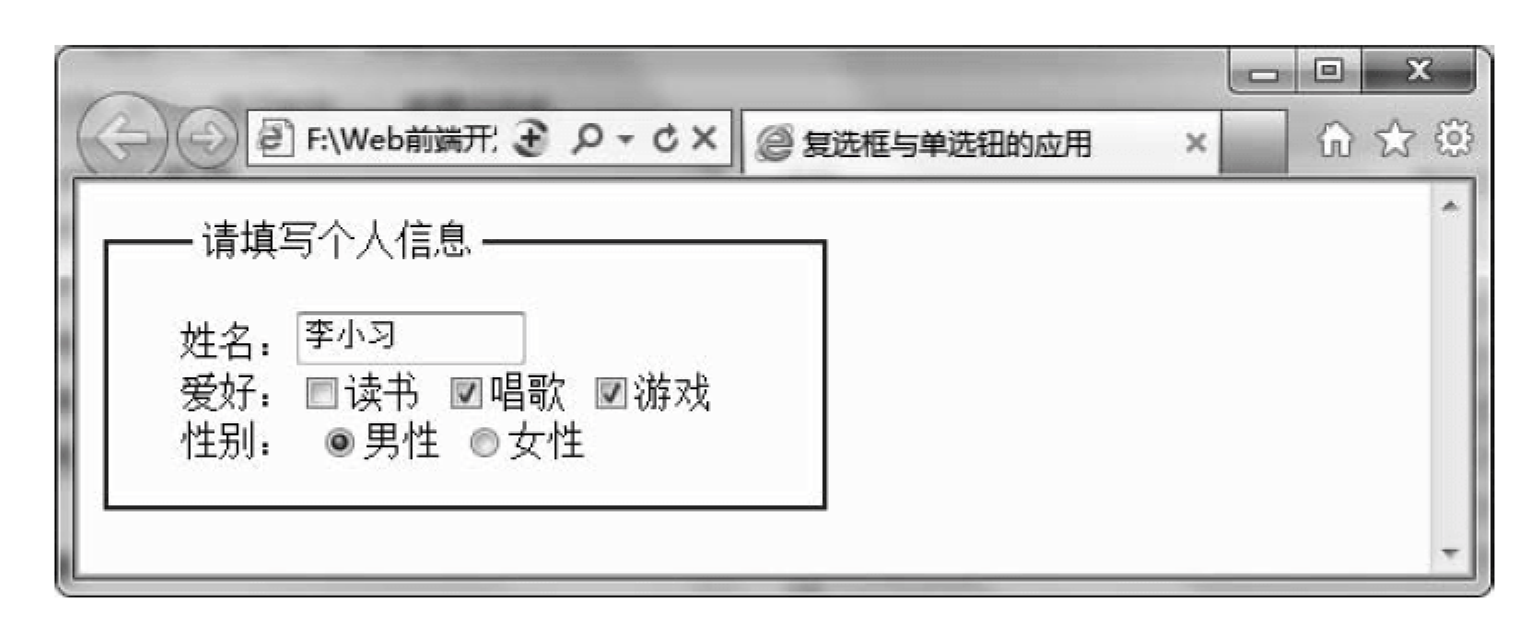



图 12-4　复选框与单选按钮的应用

## 12.3.5　图像按钮

设置 input 标记的 type 属性值为 image，可以实现向表单中插入一个图像按钮，用户可利用图像按钮在网页中插入一张图像，通过 src 属性加载图像。

### 1．基本语法

```
<input name="" type="image" src="" width="" height=""/>
```

### 2．属性说明

图像按钮主要属性有 name、src、width、height，其属性、取值及说明如表 12-7 所示。

**表 12-7　图像按钮属性、取值及说明**

| 属　性 | 值 | 说　明 |
|---|---|---|
| name | name | 定义 input 标记的名称 |
| src | URL | 定义以提交按钮形式显示的图像的 URL |
| width | width | 规定图像的宽度，单位为像素 |
| height | height | 规定图像的高度，单位为像素 |

**【例 12-3-3】**在网页中使用图像按钮。代码如下所示，页面效果如图 12-5 所示。

视频讲解

```
<!-- edu_12_3_3.html -->
<!doctype html>
<html lang="en">
    <head>
        <meta charset="UTF-8">
        <title>图像按钮实例</title>
        <style type="text/css">
            body{text-align:center;}
            input{width:150px;height:120px;}
        </style>
    </head>
    <body>
        <form>
            <h3>我国首艘航母辽宁号</h3>
                <input type="image" name="image" src="liaoninghao.jpg" align="center"/>
            <input type="submit" value="提交">
```

```
17         </form>
18     </body>
19 </html>
```

**3．代码解释**

代码第 8 行设置 body 标记样式为内容居中；第 9 行设置 input 标记宽度和高度；第 15 行在表单中插入一个图像按钮，名称为 image，图像来源路径为当前目录下的 liaoninghao.jpg；第 16 行插入一个提交按钮。当用户单击图像按钮时，URL 中会显示当前鼠标的坐标位置值（如 edu_12_3_3.html?image.x=76&image.y=69）。




图 12-5　图像按钮实例

## 12.3.6　提交按钮

设置 input 标记的 type 属性值为 submit，可以实现向表单中插入一个提交按钮，提交按钮用于将表单的信息提交至服务器进行处理。

**1．基本语法**

```
<input name="" type="submit" value="提交"/>
```

**2．属性说明**

提交按钮的属性主要有 name、value，其属性、取值及说明如表 12-8 所示。

**表 12-8　提交按钮属性、取值及说明**

| 属　　性 | 值 | 说　　明 |
|---|---|---|
| name | name | 定义 input 标记的名称 |
| value | value | 规定 input 标记的值 |

在表单中插入提交按钮时，如果不设置属性 value 的值，它的初始值是“提交查询按钮”。所以一定要给 value 属性赋值。

## 12.3.7　重置按钮

设置 input 标记的 type 属性值为 reset，可以实现向表单中插入一个重置按钮，重置按钮用于将表单中所有的输入信息清空，然后让用户可以重新填写。

**1．基本语法**

```
<input name="" type="reset" value="">
```

### 2．属性说明

重置按钮的属性主要有 name 和 value，其属性、取值及说明如表 12-9 所示。

表 12-9　重置按钮属性、取值及说明

| 属　　性 | 值 | 说　　明 |
|---|---|---|
| name | name | 定义 input 标记的名称 |
| value | value | 规定 input 标记的值 |

## 12.3.8　普通按钮

设置 input 标记的 type 属性值为 button，可以实现向表单中插入一个普通按钮。普通按钮在网页设计非常有用，如果不通过表单提交按钮来处理事件，则可以给普通按钮绑定事件代码，来实现所需的功能。

### 1．基本语法

```
<input name="" type="button" value="" onclick=""/>
```

### 2．属性说明

普通按钮的属性有 name、value 和 onclick，其属性、取值及说明如表 12-10 所示。

表 12-10　普通按钮属性、取值及说明

| 属　　性 | 值 | 说　　明 |
|---|---|---|
| name | name | 定义 input 标记的名称 |
| value | value | 规定 input 标记的值 |
| onclick | 事件代码 | 绑定事件代码、自定义函数或直接使用脚本代码 |

**【例 12-3-4】**三种按钮的应用。代码如下所示，页面效果如图 12-6 所示。

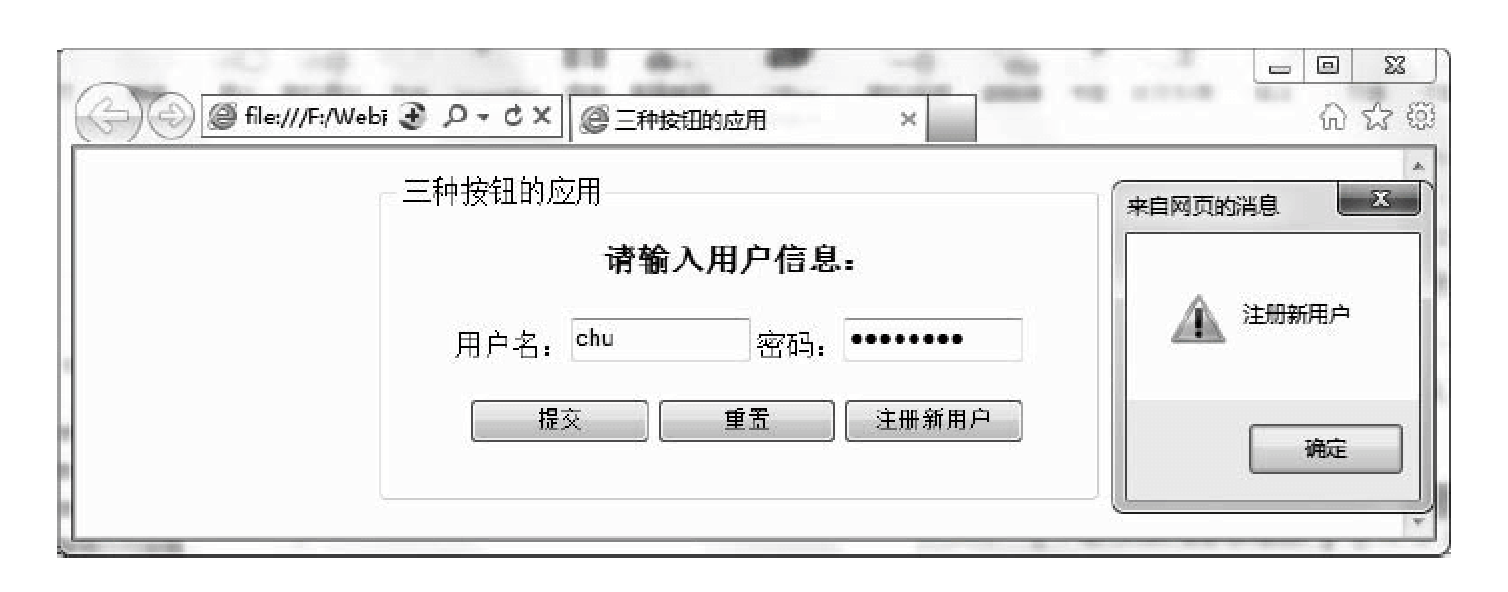



图 12-6　三种按钮的应用

```
<!-- edu_12_3_4.html -->
<!doctype html>
<html lang="en">
    <head>
        <meta charset="UTF-8">
        <title>三种按钮的应用</title>
        <style type="text/css">
            input{width:100px;height:25px;}
            body{text-align:center;}
            fieldset{width:400px;height:180px;}
        </style>
```



视频讲解

```
12    </head>
13    <body>
14        <form>
15            <fieldset>
16                <legend>三种按钮的应用</legend>
17                <h3>请输入用户信息：</h3>
18                用户名：<input type="text" name="username" size="10"/>
19                 密码：<input type="password" name="password" size="10"/><br/><br>  
20                <input type="submit" name="submit" value="提交"/>
21                <input type="reset" name="reset" value="重置"/>
22                <input type="button" name="button" value="注册新用户" onclick="javascript:alert('注册新用户');"/>
23            </fieldset>
24        </form>
25    </body>
26 </html>
```

**3．代码解释**

代码中第 8～10 行分别定义了 input、body、fieldset 标记的样式；第 15～23 行在表单中插入域和域标题标记，对表单元素进行分组，第 20 行在表单中插入一个提交按钮，名称为 submit，值为“提交”；第 21 行在表单中插入一个重置按钮，名称为 reset，值为“重置”；第 22 行在表单中插入一个普通按钮，名称为 button，值为“注册新用户”。选择“注册新用户”按钮后，触发 onclick 事件，执行 JavaScript 代码，弹出“注册新用户”告警框，如图 12-6 所示。

## 12.3.9 文件选择框

设置 input 标记的 type 属性值为 file，可以实现向表单中插入一个文件选择框。

**1．基本语法**

```
<input name="" type="file">
```

**2．属性说明**

name：定义 input 标记的名称。页面上会自动添加一个文本输入框和一个“浏览…”按钮。单击“浏览…”按钮可以从“选择要加载的文件”对话框选择某一个文件，然后将文件名称回填到文本输入框中，但并没有做任何其他操作。

## 12.3.10 隐藏框

设置 input 标记的 type 属性值为 hidden，可以实现向表单中插入一个隐藏框，用户提交表单时，隐藏框的信息也会一起提交到服务器，但隐藏框在网页中是不可见的。

**1．基本语法**

```
<input name="" type="hidden" value="" />
```

**2．属性说明**

隐藏框的属性有 name 和 value，其属性、取值及说明如表 12-11 所示。

表 12-11　隐藏框属性、取值及说明

| 属　　性 | 值 | 说　　明 |
|---|---|---|
| name | name | 定义 input 标记的名称 |
| value | value | 规定 input 标记的值 |

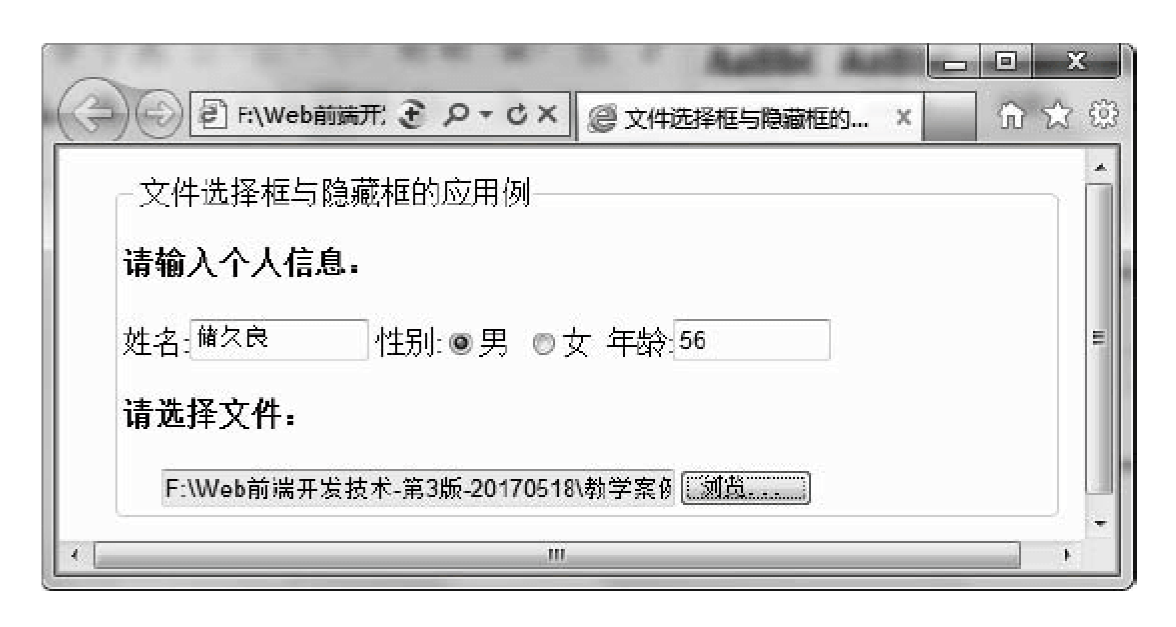
【例 12-3-5】文件选择框与隐藏框的应用。代码如下所示，页面效果如图 12-7 所示。

视频讲解

```
1 <!-- edu_12_3_5.html -->
2 <!doctype html>
3 <html lang="en">
4     <head>
5         <meta charset="UTF-8">
6         <title>文件选择框与隐藏框的应用</title>
7         <style type="text/css">
8             fieldset{width:500px;height:200px;margin:20px;}
9         </style>
10    </head>
11    <body>
12        <form>
13            <fieldset>
14                <legend>文件选择框与隐藏框的应用</legend>
15                <h4>请输入个人信息：</h4>
16                姓名:<input type="text" name="name" size="10"/>
17                性别:<input type="radio" name="sex" value="male"/>男
18                <input type="radio" name="sex" value="female"/>女 
19                年龄:<input type="text" name="age" size="8"/><br/>
20                <h4>请选择照片文件：</h4>
21                <input type="file" name="file"><br>
22                <input type="hidden" name="admin" value="ABCD">
23            </fieldset>
24        </form>
25    </body>
26 </html>
```

### 3．代码解释

代码中第 8 行定义了 fieldset 标记的样式；第 12～24 行插入表单；并在表单中插入域和域标题标记；第 16 行、第 19 行在表单中分别插入一个单行文本输入框；第 17 行、第 18 行分别插入一个单选按钮；第 21 行插入一个文件选择框，名称为 file，用户可选择相关文件。选择“浏览…”按钮后，弹出“选择要添加的文件”对话框，如图 12-7 右图所示，选择 edu_12_2_1.html 后，单击“打开”按钮，所选文件的名称自动回填到文本输入框内。



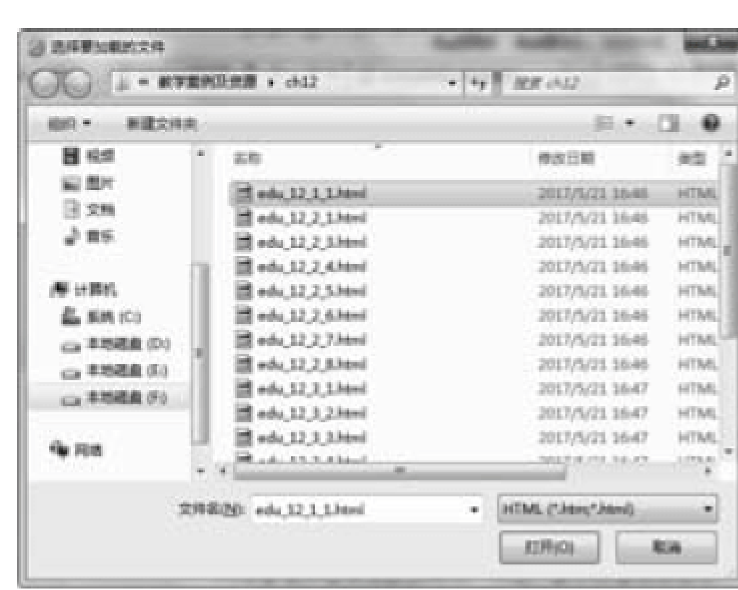



图 12-7　文件选择框与隐藏框的应用

# 12.4　多行文本输入框

网站管理员经常需要收集用户对某一事件的看法或征求一下用户的意见，而用户的反馈意见往往比较长，而单行文本输入框不能满足这一要求。textarea 标记可以向表单中插入多行文本输入框。多行文本输入框可以用来输入较多的文字信息，而且可以换行，并将这些信息提交到服务器。

## 1．基本语法

```
<textarea name="" rows="" cols="" wrap=""/>初始信息内容</textarea>
<textarea rows="" cols="" wrap="soft|hard"></textarea><!-- HTML5定义 -->
```

## 2．属性说明

多行文本输入框 textarea 标记是成对标记，其主要属性有 name、rows、cols、wrap 等，其属性、取值及说明如表 12-12 所示。默认情况下，当用户在文本区域中输入文本后，浏览器会将它们按照输入时的状态发送给服务器。只有在用户按下 Enter 键的地方生成换行。

表 12-12　多行文本输入框属性、取值及说明

| 属　性 | 值 | 说　明 |
|---|---|---|
| name | name | 定义 textarea 标记的名称 |
| rows | number | 规定文本区内的可见行数 |
| cols | number | 规定文本区内的可见宽度 |
| wrap | wrap\|virtual\|physical\|off | wrap 属性规定当在表单中提交时，文本区域中的文本如何换行<br>wrap—文本区会包含一行文本，用户必须将光标移动到右边才能看到全部文本，这时将把一行文本传送给服务器；virtual—将实现文本区内的自动换行，但在传输给服务器时，文本只在用户按下 Enter 键的地方进行换行，其他地方没有换行的效果；physical——将实现文本区内的自动换行，并以这种形式传送给服务器；off—不会自动换行，输入内容超出文本域右界时，文本将向左滚动，必须按 Enter 键才能将插入点移到下一行。HTML5 中，soft 表示提交时不换行，hard 表示提交时换行 |

【例 12-4-1】征求意见表。代码如下所示，页面效果如图 12-8 所示。

视频讲解

```
<!-- edu_12_4_1.html -->
<!doctype html>
<html lang="en">
    <head>
        <meta charset="UTF-8">
        <title>征求意见表</title>
    </head>
    <body>
        <form>
            <h3>请您填写宝贵意见：</h3>
            <textarea name="info" rows="4" cols="50" wrap="virtual">
            </textarea>
        </form>
    </body>
</html>
```

### 3．代码解释

代码中第 11 行在表单中插入了一个 4 行 50 列的多行文本输入框，名称为 info， wrap 值设为 virtual，即文本区自动换行。




图 12-8　多行文本输入框实例

## 12.5　下拉列表框

下拉列表可以在表单中接收用户的输入。下拉列表通常需要同时使用 select 和 option 标记来在表单中插入下拉菜单和列表项。

### 1．基本语法

```
 select name="" size="" multiple>
   <option value="" selected>文字信息</option>
   <option value=""></option>
   ...
</select>
```

### 2．属性说明

select 标记是成对标记，option 标记是单个标记，但应该把它补成成对标记，结构更为清晰。select 标记有 name、size、multiple 等属性。option 标记有 value、selected 等属性。select 标记与 option 标记必须配合使用。每一选项必须指定一个显示的文本和一个 value 值，显示文本通常附在 option 标记后面。它们的属性、取值及说明如表 12-13 所示。

**表 12-13　select 和 option 标记属性、取值及说明**

| 标记名称 | 属　性 | 值 | 说　明 |
|---|---|---|---|
| select | name | name | 定义 select 标签的名称 |
| | size | number | 规定下拉列表框中可见选项的数目 |
| | multiple | multiple | 规定可选择多个选项 |
| option | value | value | 规定列表项的值 |
| | selected | selected | 设置预选列表项 |

**【例 12-5-1】**下拉列表框的应用。代码如下所示，页面效果如图 12-9 所示。



视频讲解

```
<!-- edu_12_5_1.html -->
<!doctype html>
<html lang="en">
 <head>
  <meta charset="UTF-8">
        <title>下拉列表框的应用</title>
    </head>
    <body>
```

```
9         <form>
10            <h3>请选择您的课程:</h3>
11            <select name="course" size="4"  multiple>
12                <option value="c1" selected>C/C++程序设计</option>
13                <option value="c2">计算机网络</option>
14                <option value="c3" >数据结构</option>
15                <option value="c4">Java程序设计</option>
16                <option value="c5">计算机组成原理</option>
17            </select>
18        </form>
19    </body>
20 </html>
```

**3．代码解释**

代码中第 11～17 行插入了一个下拉列表框，名称为 course，选项数目为 4，设置 multiple 属性支持多选；第 12～16 行插入了 5 个列表项，列表项内容为课程名称，其中第 12 行设置 selected 属性，使列表项“C/C++程序设计”为默认选择项。

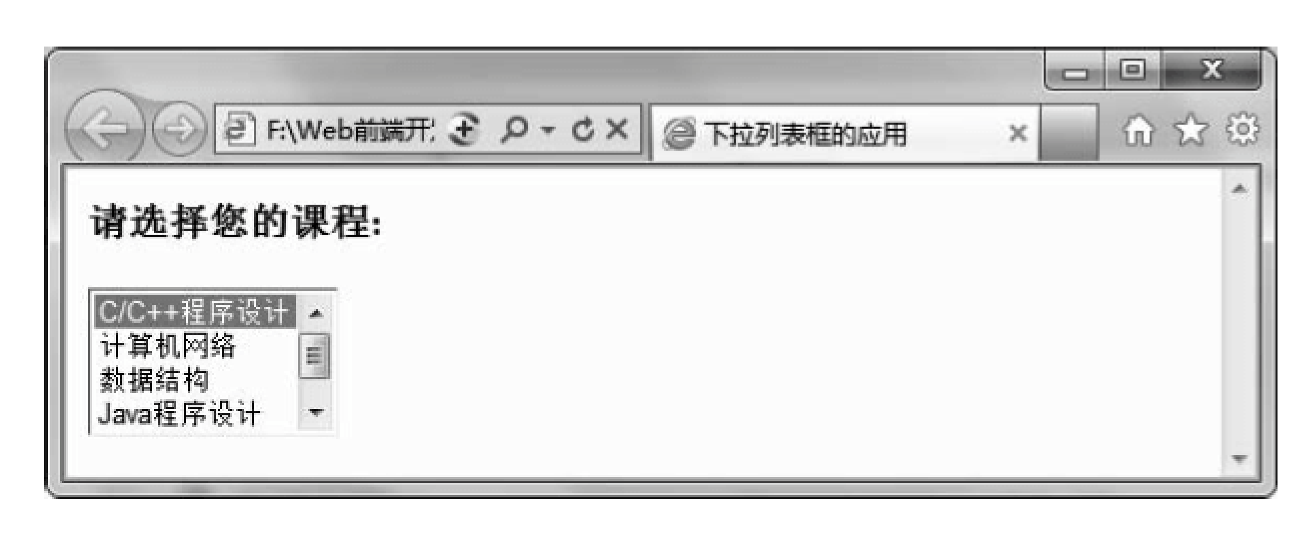



图 12-9　下拉列表框实例

# 12.6　综 合 实 例

以“第十八届中国国际广告节会议注册表”页面为例，其页面效果如图 12-10 所示。

采用 11 行 9 列的表格布局来完成页面设计，注册界面使用表单和表单控件来实现。实现的代码如下所示：




图 12-10　第十八届中国国际广告节会议注册表效果图

```
1 <!-- edu_12_6_1.html -->
2 <!doctype html>
```

```
<html lang="en">
    <head>
        <meta charset="UTF-8">
        <title>第十八届中国国际广告节会议注册表</title>
        <style type="text/css">
            body{text-align:center;}
            h1{font-size:25px;text-align:center;}
            .zhuce{font-size:14px;text-align:center;width:840px; margin: 0 auto;background:#f7f7f7;}
            .zhuce td{border:1px solid #3300cc;padding:2px 3px;}
            .zhuce .ibg{text-align:left;}
            .zhuce .bbg{padding:10px 0;font-size:13px;}
            #bt{width:100px;height:35px;background:#99ffcc;}
        </style>
    </head>
    <body>
        <h1>第十八届中国国际广告节会议注册表</h1>
        <form>
            <table class="zhuce">
                <tr>
                    <td width="100px">参会者姓名</td>
                    <td colspan="4" class="ibg">
                        <input name="txtName" type="text">
                    </td>
                    <td>职务</td>
                    <td colspan="3" class="ibg">
                        <input name="txtZhiwu" type="text">
                    </td>
                </tr>
                <tr>
                    <td>工作单位</td>
                    <td colspan="8" class="ibg">
                            <input name="txtDanwei" type="text" style= "width: 500px;">
                    </td>
                </tr>
                <tr>
                    <td>电话</td>
                    <td colspan="2" class="ibg">
                        <input name="txtTel" type="text">
                    </td>
                    <td>传真</td>
                    <td class="ibg">
                        <input name="txtFax" type="text">
                    </td>
                    <td colspan="3">手机</td>
                    <td class="ibg">
                        <input name="txtMobil" type="text">
                    </td>
                </tr>
                <tr>
                    <td>通讯地址</td>
                    <td colspan="6" class="ibg">
                        <input name="txtAddress" type="text" style= "width: 400px;">
                    </td>
                    <td>邮编</td>
                    <td class="ibg">
```

```
                <input name="txtPostCode" type="text">
            </td>
        </tr>
        <tr>
            <td>E-mail</td>
            <td colspan="6" class="ibg">
                <input name="txtEmail" type="text" style= "width: 180px;">
            </td>
            <td>国家</td>
            <td class="ibg">
               <select name="ddlCountry" id="ddlCountry" style= "width: 180px;">
                    <option value="中国" selected >中国</option>
                    <option value="欧洲-英国">欧洲-英国</option>
                    <option value="南美洲-巴西">南美洲-巴西</option>
                    <option value="美国">美国</option>
                    <option value="非洲-南非">非洲-南非</option>
             </select>
            </td>
        </tr>
        <tr>
            <td>省份</td>
            <td colspan="6" class="ibg">
                <select name="ddlProvince" style="width:180px;">
                    <option value="请选择">请选择</option>
                    <option value="北京市">北京市</option>
                    <option value="天津市">天津市</option>
                    <option value="重庆市">重庆市</option>
                    <option value="上海市">上海市</option>
                </select>
            </td>
            <td>城市</td>
            <td class="ibg">
                <input name="txtCity" type="text" style="width: 180px;">
            </td>
        </tr>
        <tr>
            <td colspan="9"><p>会议费标准（人民币）</p></td>
        </tr>
        <tr>
            <td colspan="2">身份 / 时间</td>
            <td colspan="4">2011年9月20日之前注册</td>
            <td colspan="3">2011年9月20日之后注册</td>
        </tr>
        <tr>
            <td colspan="2">中广协会员</td>
            <td colspan="4">
                <input type="radio" name="rbMem" value="rbMem1">1500元
            </td>
            <td colspan="3">
                <input type="radio" name="rbMem" value="rbMem2">1800元
            </td>
        </tr>
        <tr>
            <td colspan="2">非会员</td>
            <td colspan="4">
```

```
113                     <input type="radio" name="rbMem" value= "rbNoMem1">1800元
114                   </td>
115                   <td colspan="3">
116                     <input type="radio" name="rbMem" value="rbNoMem2">2000元
117                   </td>
118               </tr>
119               <tr>
120                   <td colspan="9" class="bbg">
121                       <input id="bt" type="submit" name="btnOk" value="提交">
122                       <input id="bt" type="reset"><br><br>
123             <a href="邀请函和注册表2011.doc">第十八届中国国际广告节注册表下载</a>
124                   </td>
125               </tr>
126           </table>
127       </form>
128   </body>
129 </html>
```

**代码解释**

代码中第 17～127 行在 HTML 的 body 标记中插入表单，在表单中又插入一个 11 行 9 列的表格；

第 24 行、28 行、34 行、40 行、44 行、48 行、54 行、58 行、64 行插入单行文本输入框，分别用于输入参会者姓名、职务、工作单位、电话、传真、手机、通讯地址、邮编、E-mail 等信息；

第 68～74 行插入下拉列表框，用于输入用户所属国家，中国为预选状态；

第 80～86 行插入下拉列表框，用于输入用户所属省份；

第 104 行、107 行、113 行、116 行插入单选按钮，输入会员信息和缴费信息；

第 121 行、第 122 行分别插入提交按钮和重置按钮，用于提交整个表单信息和清空表单内容。

# 本 章 小 结

表单是 Web 服务器端和客户端进行信息交互的主要桥梁。Web 服务器通过含有表单和表单控件的 Web 页面完成用户信息的采集。表单有三个重要属性，分别是 name、action、method。表单有 12 个常用表单控件，分别是单行文本输入框、密码输入框、复选框、单选按钮、图像按钮、提交按钮、重置按钮、普通按钮、文件选择框、隐藏框、多行文本输入框、下拉列表框。使用域和域标题可以对表单元素进行合理分组。组合运用这些标记，可以使 HTML 网页和用户更加灵活地交互信息。

# 练习与实验

## 练 习 12

**1．选择题**

（1）下列选项不是表单标记的属性是（　　）。

A．method　　B．action　　C．enctype　　D．option

（2）下列选项不是 input 标记的 type 属性值的是（　　）。

A．password　　B．radio　　C．textarea　　D．button

（3）下列 input 标记的类型属性取值表示复选框的是（　　）。

A．hidden　　B．checkbox　　C．radio　　D．select

（4）下列 input 标记的类型属性取值表示单选按钮的是（　　）。

A．hidden　　B．checkbox　　C．radio　　D．select

（5）用于设置文本输入框显示宽度的属性是（　　）。

A．size　　B．maxlength　　C．value　　D．length

**2．填空题**

（1）表单 form 标记中，method 属性的取值可以为________和________。

（2）表单是 Web________和 Web________之间实现信息交流和传递的桥梁。

（3）<select>标记必须与________标记配合使用，包含________、________和________属性。

（4）________标记用于定义多行文本输入框，指定行数的属性为________，指定列数的属性为________。

（5）重置按钮的 type 属性值为________，提交按钮的 type 属性值为________，普通按钮的 type 属性为________。

（6）一组复选框中复选框的 name 属性值必须_______，value 值也必须_______；而一组单选按钮中每一个单选按钮的 name 属性值必须_____，value 属性值必须_______。

（7）通过______属性可以将某一复选框、单选按钮设置为默认预选状态；通过_______属性以将下拉列表框中的某一选项设置为默认预选状态。

（8）使用______标记可以定义域，使用______标记可以定义域的标题。

## 实验 12

1．编写程序实现如图 12-11 所示的登录页面。

2．利用表单和表单元素设计简单的应聘页面，如图 12-12 所示，写出实现的 HTML 代码。

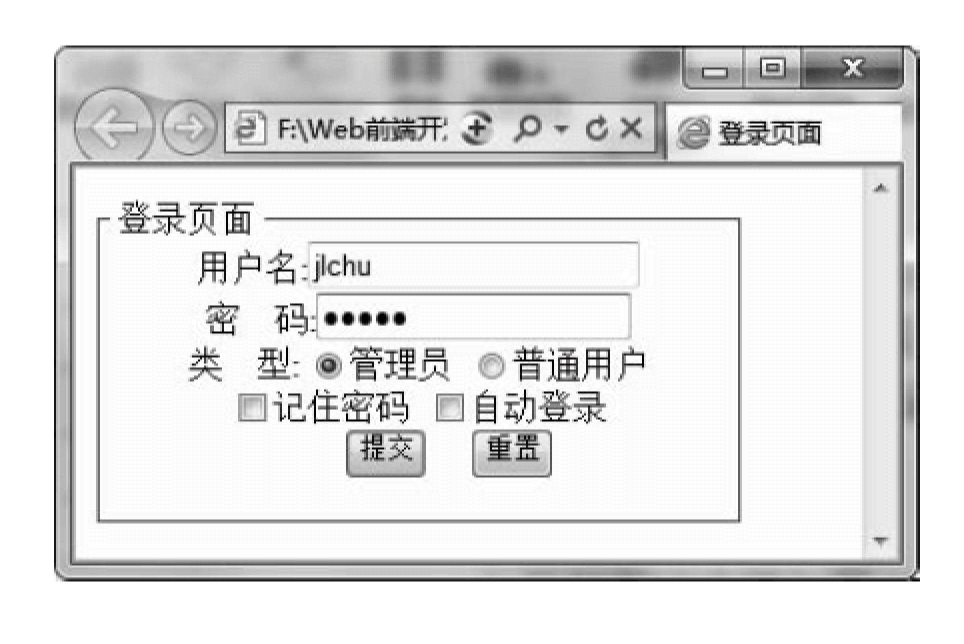



图 12-11　登录页面效果图




图 12-12　应聘页面效果图

# 第 13 章 HTML5 基础与 CSS3 应用

**本章学习目标**

随着移动互联技术不断发展，HTML5、CSS3、JavaScript、jQuery Mobile 等技术在移动端应用越来越普及，所以学会移动开发技术已经是势在必行了。本章主要简单地介绍 HTML5 的新特点、新增加标记及新增属性，学会运用 CSS3 新特性来改变网页的外在表现，以增加用户的体验。

Web 移动前端开发工程师应掌握以下内容:

- 熟悉掌握 HTML5 新特性。
- 掌握 HTML5 页面结构。
- 学会使用 HTML5 新增元素和新增属性。
- 掌握 HTML5 新增表单元素及新增属性的设置方法。
- 学会使用 HTML5 的 Audio 和 Video 媒体元素。
- 学会 HTML5 本地存储开发简易 Web 应用。
- 学会使用 CSS3 的转换、过渡和动画等特性设计页面的动态效果。
- 学会设置与应用 CSS3 文本效果及多列等属性。

W3C（World Wide Web Consortium, 万维网联盟）自 2008 年 1 月 22 日公布 HTML5 草案到 2014 年 10 月 28 日发布正式标准，历时多年终于完成标准的制定。目前 HTML5 已经成为 HTML、XHTML 及 HTML DOM 的新标准，大多数浏览器已经支持 HTML5 技术。2016 年 11 月 1 日 W3C 正式发表了 HTML5.1 推荐标准，该推荐标准定义了 HTML 语言第 5 个版本的第 1 个小版本（https://www.w3.org/TR/2016/REC-html51-20161101/），绝大多数的主流浏览器已经能够实现或即将实现 HTML5.1 引入的新特性和变化，同时 W3C 已在着手制定下个版本 HTML5.2。

## 13.1 HTML5 概述

由于 HTML4.01 标准的标记能力不足，而 XHTML1.0 标准又过于严格、兼容性差，实际 Web 应用开发中很难完全遵守 W3C 所制定的规范。Web 应用开发中面临许多困境，如很多人开始怀疑 Flash 的安全性等问题，但又找不到合适的插件，Web 前端开发工程师纷纷埋怨在开发 PC 端和移动端应用时，仍然需要为微软、苹果、安卓等系统设计不同方案。2004 年，为了推动 Web 标准化运动的发展，由 Apple、Opera、Google、Mozilla 等公司发起，与一些浏览器生产厂家和相关团体共同成立一个协作组织，称之为 WHATWG（Web Hypertext Application Technology Working Group，Web 超文本应用技术工作组），WHATWG

组织专门致力于 Web 表单和应用程序，当时 W3C 专注于 XHTML 2.0 标准的制定。2006 年 10 月，W3C 决定与 WHATWG 合作共同研制 HTML5 相关技术标准。

在 HTML5 中需要弄清楚元素、标记和属性的相关概念，以便于正确理解和阅读本章的内容。标记就是被尖括号“<”和“>”包围起来的关键字，表示特定功能的符号。绝大部分的标记都是双（成对）标记，如<html></html>、<head></head>等。少部分是单(个)标记，如<br>、<hr>、<meta>、<link>等。标记就是用来说明 HTML 元素的。一个非空 HTML 元素是由开始标记、元素的属性和值、内容和结束标记组成的，是构成 HTML 文件的基本对象。位于起始标记和结束标记之间的文本就是 HTML 元素的内容。为 HTML 元素提供各种附加信息的就是 HTML 属性，它总是以属性名="属性值"这种名值对的形式出现，而且属性总是在 HTML 元素的开始标记中进行定义。示范代码如下所示：

```
<html>
  <head>                                        <!-- 这是开始标记 -->
    <title>元素、标记和属性讲解</title>          <!-- 这是一个title元素 -->
   </head>                                      <!-- 这是结束标记 -->
  <body onload="alert('页面装载!');">          <!-- 这是在开始标记内定义属性并赋值 -->
    <h3>这是元素的内容</h3>                      <!-- 这是一个h3元素 -->
  </body>
</html>
```

在这个示例中，“<h3>这是元素的内容</h3>”就是 HTML 元素，其中“这是元素的内容”就是元素的具体内容。<head>、<title>、<body>等就是 HTML 标记，这些标记构成了 HTML 元素。<body onload="alert('页面装载！');">中的 onload="alert('页面装载！');"就是标记的属性，总之，元素和标记的区别也不必太在意，实际工作中都直接以标记统称。而属性就是为 HTML 标记添加各种附加信息或者配置选项的参数。

## 13.1.1 HTML5 的八个特性

HTML5 具有八个特性，分别对应八种 logo（https://www.w3.org/html/logo/#downloads），如图 13-1 所示，左边为 HTML5 的新 logo，右边为 HTML5 八个特性的 logo。




图 13-1　HTML5 的新 logo

**1．语义特性（Semantic）**

HTML5 赋予网页更好的意义和结构。更加丰富的标记集将随着对 RDFa（RDF attribute）、微数据与微格式等方面的支持，构建对程序、用户都更有价值的数据驱动的 Web。

**2．离线与存储特性（Offline & Storage）**

基于 HTML5 开发的网页 APP 拥有更短的启动时间，更快的联网速度，这些全得益于

HTML5 APP Cache、本地存储功能、Indexed DB 和 File API 说明文档。

**3．设备访问特性 (Device Access)**

从 Geolocation 功能的 API 文档公开以来，HTML5 为网页应用开发者们提供了更多功能上的优化选择，带来了更多体验功能的优势。HTML5 提供了前所未有的数据与应用接入开放接口。使外部应用可以直接与浏览器内部的数据相连，例如视频影音可直接与麦克风及摄像头相连。

**4．多媒体特性(Multimedia)**

支持网页端的 Audio、Video 等多媒体功能，与网站自带的 APPS、摄像头、影音功能相得益彰。

**5．三维、图形与特效特性（3D、Graphics & Effects）**

基于 SVG、Canvas、WebGL 及 CSS3 的 3D 功能，用户会惊叹于浏览器所呈现的惊人视觉效果。

**6．性能与集成特性（Performance & Integration）**

没有用户愿意一直等待——HTML5 会通过 Web Workers 和 XMLHttpRequest2 等技术，帮助 Web 应用和网站在多样化的环境中更快速地工作。

**7．连接特性（Connectivity）**

更高的连接工作效率，使基于页面的实时聊天、更快速的网页游戏体验、更优化的在线交流得以实现。HTML5 拥有更有效的服务器推送技术，Server-Sent Event 和 WebSockets 就是其中的两个特性，这两个特性能够帮助我们实现服务器将数据“推送”到客户端的功能。

**8．CSS3 特性(CSS3)**

在不牺牲性能和语义结构的前提下，CSS3 中提供了更多的风格和更强的效果。此外，较之于之前的 Web 排版、Web 的开放字体格式（Web Open Font Format，WOFF）提供了更高的灵活性和控制性。

### 13.1.2 HTML5 的优势

**1．摆脱对平台的依赖**

HTML5 摆脱了对平台的依赖，用户打开浏览器，直接就可以访问自己的应用，而不需要经过各种 Store 的审核。

**2．实时更新**

实时更新，通常平台的审核都需要七个工作日左右的时间，在发布之后发现问题怎么办？Web 方式就不存在这种问题。

**3．离线使用**

用户可以离线使用，更新下载量较少，可以全部更新，也可以选择替换部分文件。

**4．代码更安全**

使用 HTML5，代码更安全。众所周知，Web 应用有一个很大的问题就是代码的安全性问题，但现在 HTML5 可以将 Web 代码全部加密，本地应用解密后再运行，大大提高了代码的安全性。

**5．跨平台**

HTML5 可以做到跨平台，大部分核心代码不需要重写，JavaScript 的代码可以在许多地方使用，包括移动应用、移动网站、PC 网站、各种浏览器插件，甚至可以用 WebKit 封装作为跨平台的应用程序。

**6．可以充分利用 Native**

HTML5 可以通过浏览器作为中介充分利用 Native 的好处，例如可以使用 GPS、照相机、本地相册、读取本地联系人，也可以使用推送功能等，最重要的是，某些 Web 无法实现的功能，可以利用 Native 来实现。

## 13.1.3 HTML5 新增结构元素及页面元素

HTML5 中新增结构元素如表 13-1 所示。

**表 13-1 HTML5 新增结构元素表**

| 元　　素 | 说　　明 |
|---|---|
| header | 页面或页面中某一个区块的页眉，通常是一些引导和导航信息 |
| nav | 可以作为页面导航的链接组 |
| section | 页面中的一个内容区块，通常由内容及其标题组成 |
| article | 代表一个独立的、完整的相关内容块，可独立于页面其他内容使用 |
| aside | 非正文的内容，与页面的主要内容是分开的，被删除而不会影响到网页的内容 |
| footer | 页面或页面中某一个区块的脚 |

HTML5 中新增页面元素如表 13-2 所示。

**表 13-2 HTML5 新增其他页面元素表**

| 元　　素 | 说　　明 |
|---|---|
| video | 定义视频。像电影片段或其他视频流 |
| audio | 定义音频。如音乐或其他音频流 |
| embed | 用来嵌入各种媒体内容。格式如 Midi、Wav、AIFF、AU、MP3 及 Flash 等 |
| mark | 主要用来在视觉上向用户呈现那些需要突出显示或高亮显示的文字 |
| progress | 表示运行中的进程，可以使用 progress 元素显示 JavaScript 中耗时时间函数的进程。等待中、请稍候等 |
| time | 表示日期或时间，也可以两者同时 |
| ruby | 定义 ruby 注释（中文注音或字符）。ruby、rt 元素一同使用。ruby 元素由一个或多个字符（需要一个解释/发音）和一个提供该信息的 rt 元素组成，还包括可选的 rp 元素，定义当浏览器不支持 "ruby" 元素时显示的内容 |
| rt | 定义字符（中文注音或字符）的解释或发音 |
| rp | 在 ruby 注释中使用，以定义不支持 ruby 元素的浏览器所显示的内容 |
| wbr | 表示软换行。br 元素表示此处必须换行；wbr 表示浏览器窗口或父级元素足够宽时（没必要换行时）不换行，而宽度不够时主动在此处换行 |
| canvas | 定义图形，例如图表和其他图像。该元素只是图形容器（画布），必须使用脚本来绘制图形 |
| command | 表示命令按钮，比如单选按钮、复选框或按钮。只有当 command 元素位于 menu 元素内时，该元素才是可见的，否则不会显示这个元素，但是可以用它规定键盘快捷键 |

续表

| 元　素 | 说　明 |
| --- | --- |
| details | 用于描述文档或文档某个部分的细节。可与 summary 元素配合使用，summary 可以为 details 定义标题。标题是可见的，用户单击标题时，会显示出 details。summary 应该是 details 的第一个子元素 |
| datalist | 定义选项列表。与 input 元素配合使用该元素，来定义 input 可能的值。datalist 及其选项不会被显示出来，它仅仅是合法的输入值列表。使用 input 元素的 list 属性来绑定 datalist |
| output | 定义不同类型的输出，例如脚本的输出 |
| source | 为媒介元素（如 video 和 audio）定义媒介资源 |
| menu | 定义菜单列表。当希望列出表单控件时使用该元素。注意与 nav 的区别，menu 专门用于表单控件 |

## 13.1.4　HTML5 废除的元素与属性

**1．HTML5 废除的元素**

HTML4.01 之前有些标记被不赞成使用，HTML5 已经淘汰了，建议使用 CSS 来替代。还有些标记 HTML5 已经不再支持，所以也需要淘汰。

（1）纯表现的元素，如 font、basefont、center、big、s、u、strike、tt。

（2）对可用性产生负面影响的元素，如 frameset、frame、noframes 等元素。HTML5 只支持浮动框架（内联框架）iframe 元素。

（3）易产生混淆的元素，如 acronym、applet、isindex、dir 等元素。

（4）废除只有部分浏览器支持的元素，如 blink、bgsound、marquee 等元素。

（5）其他被废除的元素，如废除 rb，使用 ruby 替代；废除 listing，使用 pre 替代；废除 xmp，使用 code 替代；废除 nextid，使用 guids 替代；废除 plaintex，使用 text/plian（无格式正文）MIME 类型替代。

**2．HTML5 废除的属性**

HTML4.01 中一些属性在 HTML5 中不再被使用，而是采用其他属性或其他方式进行替代。这些属性如表 13-3 所示。

**表 13-3　HTML5 废除的属性表**

| 废 除 属 性 | 隶 属 元 素 | HTML 5 替代方案 |
| --- | --- | --- |
| rev | link、a | rel |
| charset | link、a | 在被链接的资源中使用 HTTP Content-type 头元素 |
| shape、coords | a | 使用 area 元素代替 a 元素 |
| longdesc | img、iframe | 使用 a 元素链接到较长描述 |
| target | link | 多余属性，被省略 |
| nohref | area | 多余属性，被省略 |
| profile | head | 多余属性，被省略 |
| version | html | 多余属性，被省略 |
| name | img | id |
| scheme | meta | 只为某个表单域使用 scheme |

续表

| 废除属性 | 隶属元素 | HTML 5 替代方案 |
| --- | --- | --- |
| archive、chlassid、codebose、codetype、declare、standby | object | 使用 data 与 type 属性类调用插件。需要使用这些属性来设置参数时，使用 param 属性 |
| valuetype、type | param | 使用 name 与 value 属性，不声明其 MIME 类型 |
| axis、abbr | td、th | 使用以明确简洁的文字开头、后跟详述文字的形式。可以对更详细内容使用 title 属性，来使单元格的内容变得简短 |
| scope | td | 在被链接的资源中使用 HTTP Content-type 头元素 |
| align | caption、input、legend、div、h1、h2、h3、h4、h5、h6、p | 使用 CSS 样式表替代 |
| alink、link、text、vlink、background、bgcolor | body | 使用 CSS 样式表替代 |
| align、bgcolor、border、cellpadding、cellspacing、frame、rules、width | table | 使用 CSS 样式表替代 |
| align、char、charoff、height、nowrap、valign | tbody、thead、tfoot | 使用 CSS 样式表替代 |
| align、bgcolor、char、charoff、height、nowrap、valign、width | td、th | 使用 CSS 样式表替代 |
| align、bgcolor、char、charoff、valign | tr | 使用 CSS 样式表替代 |
| align、char、charoff、valign、width | col、colgroup | 使用 CSS 样式表替代 |
| align、border、hspace、vspace | object | 使用 CSS 样式表替代 |
| clear | br | 使用 CSS 样式表替代 |
| compace、type | ol、ul、li | 使用 CSS 样式表替代 |
| compace | dl | 使用 CSS 样式表替代 |
| compace | menu | 使用 CSS 样式表替代 |
| width | pre | 使用 CSS 样式表替代 |
| align、hspace、vspace | img | 使用 CSS 样式表替代 |
| align、noshade、size、width | hr | 使用 CSS 样式表替代 |
| align、frameborder、scrolling、marginheight、marginwidth | iframe | 使用 CSS 样式表替代 |
| autosubmit | menu | |

### 13.1.5 浏览器支持与选择

一些低版本的浏览器并不支持 HTML5，如 IE6～IE8 浏览器。所有新、旧浏览器，对

无法识别的元素均会视作内联(inline)元素来自动处理。可以通过其他方法让这些浏览器能够处理“未知”的 HTML 元素。

html5shiv 是针对 IE 浏览器比较好的解决方案。html5shiv 主要解决 HTML5 提出的新的元素不被 IE6～IE8 识别，这些新元素不能作为父节点包裹子元素，并且不能应用 CSS 样式的问题。通过 document.createElement(elementName)方法实现让 CSS 样式应用在未知元素上。html5shiv 就是根据这个原理创建的，使用是非常简单的。

使用 Sjoerd Visscher 创建的“HTML5 Enabling JavaScript”，即“shiv(开门闸刀)”来解决该问题。引用 Google 的 html5.js 文件的代码如下所示：

```
<head>
  <!--[if lt IE 9]>
     <script src="http://html5shiv.googlecode.com/svn/trunk/html5.js"></script>
  <![endif]-->
</head>
```

上述代码以注释方式呈现，其作用是当 IE 浏览器的版本小于 IE9 时将读取 html5.js 文件，并解析它。也可以从 https://github.com/aFarkas/html5shiv/上直接下载并保存到本地项目目录中，有多个版本可供选择。

根据需要下载并引用外部 JS 文件，代码如下所示。

```
<head>
  <!--[if lt IE 9]>
      <script src="projectPathName/html5shiv.js"></script>
  <![endif]-->
</head>
```

最后在 CSS 文件中或样式表中添加如下标记样式。代码如下所示：

```
/* html5 将新元素变成块元素 */
article,aside,dialog,header,section,footer,nav,figure,menu{displ
ay:block}
```

考虑到浏览器的兼容性和对 HTML5 的支持情况，根据 http://html5test.com 测试结果，选择支持 HTML5 最好的 Google 公司的 Chrome 浏览器作为页面效果展示浏览器。

## 13.2　HTML5 文档结构

HTML5 文档结构同样是由头部和主体两部分组成，只是新增了一些结构元素，如 header、nav、article、section、aside、footer 六个结构元素，这些元素都是块级元素。

### 13.2.1　HTML5 页面结构

HTML4.01 之前，通常使用 DIV+CSS 来进行页面布局，采用 DIV 分割页面，采用 CSS 定义 DIV 的样式，页面效果如图 13-2 所示。HTML5 中采用页眉、页脚、导航、文章内容等结构元素来进行页面布局，显得十分方便，页面效果如图 13-3 所示。

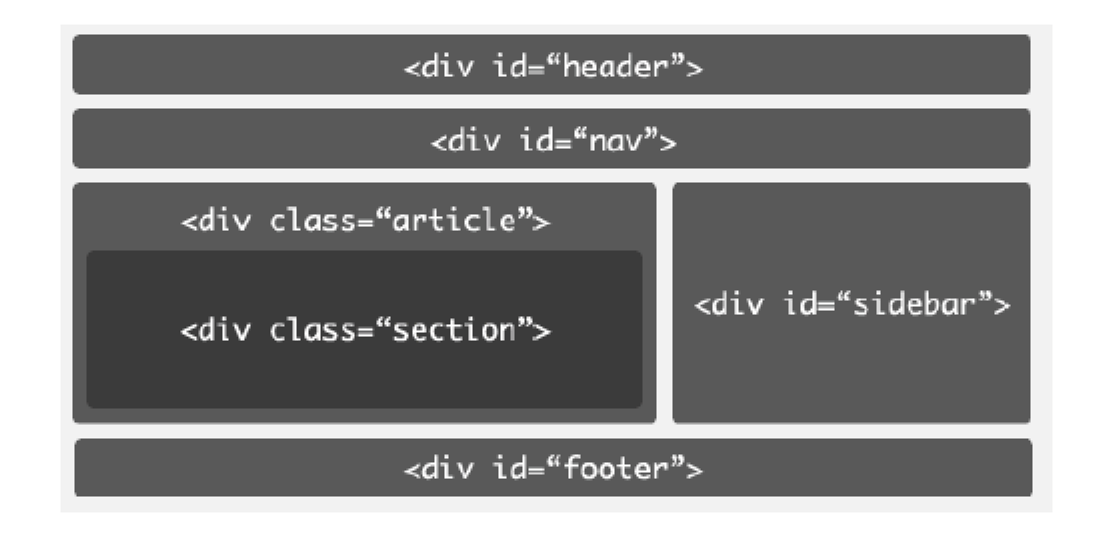



图 13-2　HTML4.01 页面布局

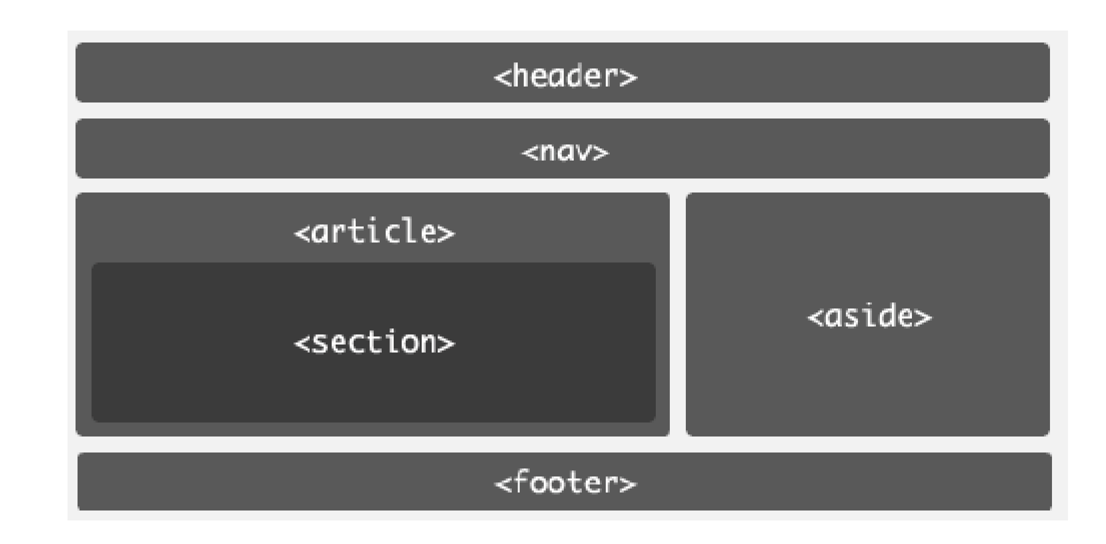



图 13-3　HTML5 结构元素布局

**HTML5 页面结构元素语法**

```
<!doctype html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="Keywords" content="">
    <meta name="Description" content="">
    <title>HTML5文档结构</title>
</head>
<body>
    <header>
      <nav>...</nav>
    </header>
    <article>
     <section>...</section>
    </article>
    <aside>...</aside>
    <footer>...</footer>
</body>
</html>
```

## 13.2.2　HTML5 新增结构元素

### 1. header 标记

header 标记定义文档和区域的页眉，通常是一些引导和导航信息。它不局限于写在网页头部，也可以写在网页内容里面。通常<header>标记至少包含（但不局限于）一个标题标记（h1～h6），也可以包括 hgroup（标题组合）标记、表格标识、搜索表单、导航等。

**【例 13-2-1】**标题组合标记的应用。代码如下，页面如图 13-4 所示。

视频讲解

```
<!-- edu_13_2_1.html -->
<!doctype html>
<html lang="en">
    <head>
        <meta charset="UTF-8">
        <title>HTML5结构元素header和hgroup标记的应用</title>
    </head>
    <body>
        <header>
            <hgroup>
                <h1>HTML5 是下一代的 HTML。</h1>
                <h3>什么是 HTML5？</h3>
                <h5>HTML5 将成为 HTML、XHTML 以及 HTML DOM 的新标准。</h5>
            </hgroup>
```

```
        </header>
    </body>
</html>
```




图 13-4　HTML5 结构元素 header 应用

**2．nav 标记**

nav 标记代表页面的一个部分，是一个可以作为页面导航的链接组。建议不要在 footer 元素中使用 nav 元素，否则易造成页面显示不正确。配置相应的 CSS 代码可以实现水平导航。

**【例 13-2-2】**导航 nav 标记的应用。代码如下，页面如图 13-5 所示。




图 13-5　HTML5 结构元素 nav 应用



视频讲解

```
<!-- edu_13_2_2.html -->
<!doctype html>
<html lang="en">
    <head>
        <meta charset="UTF-8">
        <title>HTML5结构元素header和nav标记的应用</title>
    </head>
    <body>
        <header>
            <nav>
                <ul>
                    <li><a href="#">HTML 参考手册</a></li>
                    <li><a href="#">HTML 实例</a></li>
                    <li><a href="#">HTML 测验</a></li>
                </ul>
            </nav>
        </header>
    </body>
</html>
```

**3．article 标记**

article 标记是一个特殊的 section 标记，它比 section 具有更明确的语义，它代表一个独立的、完整的相关内容块，可独立于页面其他内容使用。例如论坛帖子、博客文章、新闻故事、评论等。一般来说，article 会有标题部分，通常包含在 header 内，有时也会包含 footer。article 标记可以嵌套，内层的 article 对外层的 article 标记有隶属关系。例如一篇博客的文

章可以用 article 显示，然后后续的一些评论可以用 article 的形式嵌入其中。

【例 13-2-3】文章 article 标记的应用。代码如下，页面效果如图 13-6 所示。

视频讲解

```
<!-- edu_13_2_3.html -->
<!doctype html>
<html lang="en">
    <head>
        <meta charset="UTF-8">
        <title>HTML5结构元素artical和header标记的应用</title>
    </head>
    <body>
        <article>
            <header>
                <hgroup>
                    <h1>HTML 5结构标记的简介</h1>
                    <h2>HTML 5的诞生</h2>
                </hgroup>
                <time datetime="2017-04-28">2017-04-28</time>
            </header>
            <p>HTML5 引入了许多新标签，包括几个用于更好地描述文本结构的标签。在本文中，我们将了解这些 HTML5 引入的新的结构化标签以及如何使用它们将一个文档划分成几个内容块。</p>
        </article>
    </body>
</html>
```

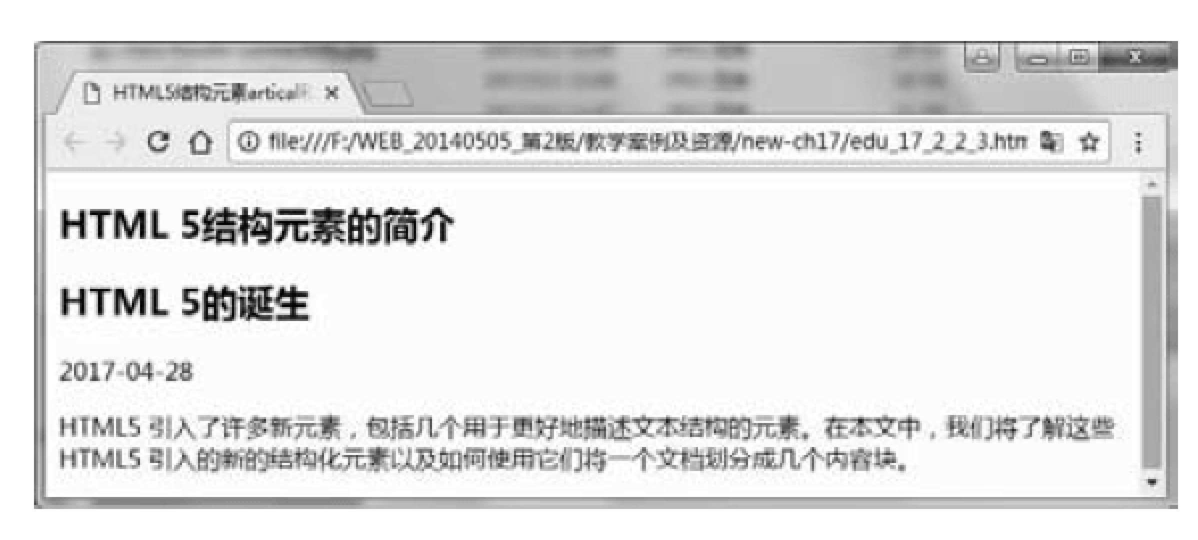



图 13-6　HTML5 结构元素 article 应用

## 4. section 标记

section 标记定义文档中的节。例如章节、页眉、页脚或文档中的其他部分。一般用于成节的内容，会在文档流中开始一个新的节。它用来表现普通的文档内容或应用区块，通常由内容及其标题组成。section 元素不是一个普通的容器元素，它表示一段专题性的内容，可以带有标题。如果描述一件具体的事物，建议使用 article 来代替 section；如果使用 section，仍可以使用 h1 作为标题，而不用担心它所处的位置。如果一个容器需要定义样式或定义行为，建议用 div 元素。

【例 13-2-4】section 标记的应用。代码如下，页面效果如图 13-7 所示。



视频讲解

```
<!-- edu_13_2_4.html -->
<!doctype html>
<html lang="en">
    <head>
        <meta charset="UTF-8">
        <title>HTML5结构元素artical和section标记的应用</title>
    </head>
    <body>
        <section>
```

```
            <h1>section标记</h1>
            <p>用来定义文档中的节(section、区段)。比如章节、页眉、页脚或文档中的其他部分。</p>
        </section>
        <section>
            <h1>article标记</h1>
            <p>article标记标识了Web页面中的主要内容。以博客为例，每篇帖子都构成一个重要内容。</p>
        </section>
    </body>
</html>
```

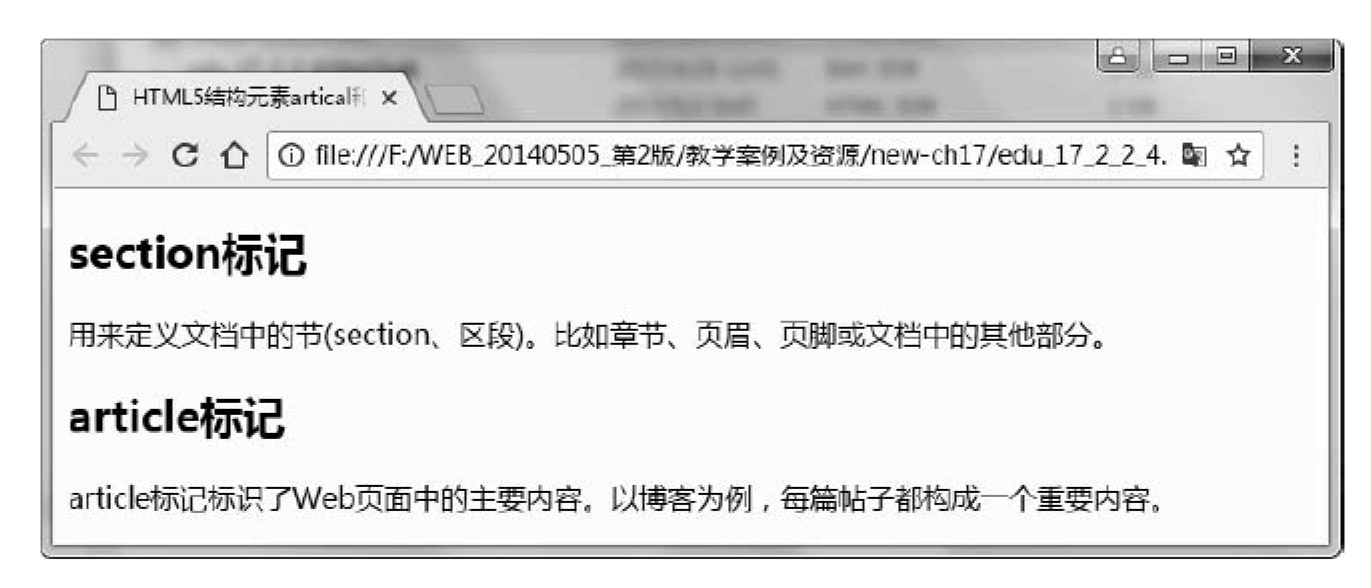



图 13-7　HTML5 结构元素 section 应用

### 5．aside 标记

aside（侧栏，也称为旁注）标记用来说明其所包含的内容与页面主要内容相关，但不是该页面的一部分，类似于使用括号对正文进行注释（就像这样）。括号中的内容提供关于该元素的一些附加信息，例如广告、成组的链接、侧栏等。

**【例 13-2-5】** aside 标记的应用。代码如下，页面效果如图 13-8 所示。

视频讲解

```
<!-- edu_13_2_5.html -->
<!doctype html>
<html lang="en">
    <head>
        <meta charset="UTF-8">
        <title>HTML5结构元素aside和artical标记的应用</title>
    </head>
  <body>
    <header>我的博客</header>
      <section>
        <article>
          <p>这是页面上重要的内容部分。也许是博客文章。带aside元素。</p>
          <aside style="float:right;width:100px;height:100px;background:#EEFFCC;">
            <p>这是第一篇博客文章。</p>
          </aside>
        </article>
        <article>
          <p>这是页面上重要的内容部分。也许是博客文章。不带aside元素</p>
        </article>
      </section>
   </body>
</html>
```

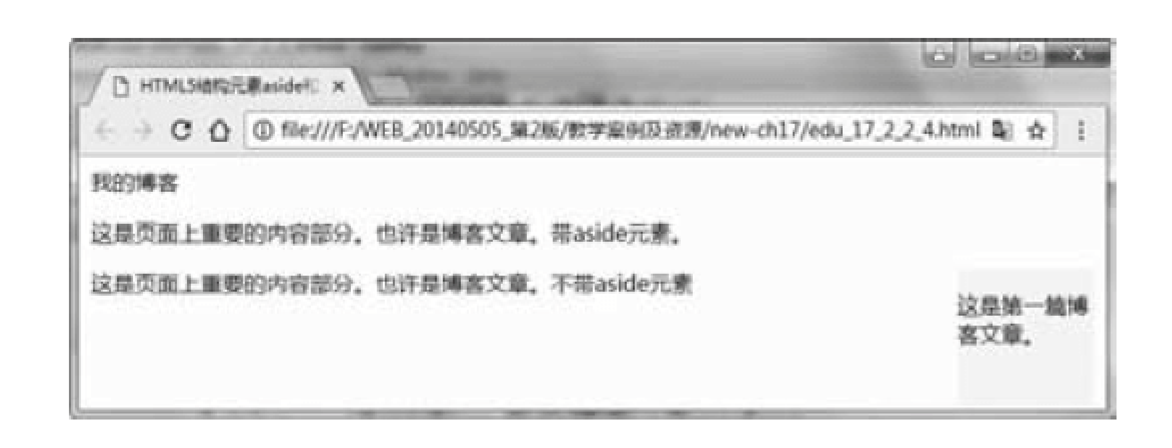



图 13-8　HTML5 结构元素 aside 应用

**6．footer 标记**

footer 标记定义 section 或文档的页脚，包含了与页面、文章或部分内容有关的信息，例如文章的作者或者日期。作为页面的页脚时，一般包含了版权、相关文件和链接。它与页眉 header 标记用法相同，在一个页面中可以多次使用，若在一个区段的最后使用 footer 标记，那么它就相当于该区段的页脚。

**【例 13-2-6】**footer 标记的应用。代码如下，页面效果如图 13-9 所示。




图 13-9　HTML5 结构元素 footer 应用

视频讲解

```
<!-- edu_13_2_6.html -->
<!doctype html>
<html lang="en">
    <head>
        <meta charset="UTF-8">
        <title>HTML5结构元素footer和section标记的应用</title>
    </head>
    <body>
        <footer>
            <div style="text-align:center;">
                <section>
                       <a href="http://www.caict.ac.cn/" target="_blank">CAICT中国信通院</a>
                    <a href="//www.w3.org/" target="_blank">W3C</a>
                    <a href="//www.dcloud.io/" target="_blank">DCloud</a>
                </section>
                <span style="padding:2px 5px;">京ICP备12046007号-5</span>
                <span style="padding:2px 5px;">HTML5中国产业联盟版权所有</span>
            </div>
        </footer>
    </body>
</html>
```

# 13.3　HTML5 新增页面元素

HTML5 除了新增的结构元素 header、nav、article、aside、section、footer 外，还增加了新的内联元素（time、meter 及 progress 等）、新的内嵌元素（video 和 audio）、新的交互

元素（details、datagrid 和 command 等）及其他页面元素。

### 13.3.1 hgroup 标记

标题组合 hgroup 标记是对网页或区段 section 的标题元素（h1～h6）进行组合。例如，在某一区段中需要连续设置多个标题标记，可以使用 hgroup 标记来组合。

【例 13-3-1】hgroup 标记的应用。代码如下，页面效果如图 13-10 所示。

视频讲解

```
<!-- edu_13_3_1.html -->
<!doctype html>
<html lang="en">
    <head>
        <meta charset="UTF-8">
        <title>HTML5页面元素hgroup标记的应用</title>
    </head>
    <body>
        <hgroup>
            <h1>JSDoc+规范</h1>
            <h2 style="color:red;">介绍</h2>
        </hgroup>
        <p style="text-indent:2em;">编写JSDoc是为了增强代码的可读性，以及方便导出API文档。它的规范可参考JSDoc 3。对于代码规范要求高的工程师和JS框架的开发者，熟悉JSDoc是必需的技能。</p>
    </body>
</html>
```

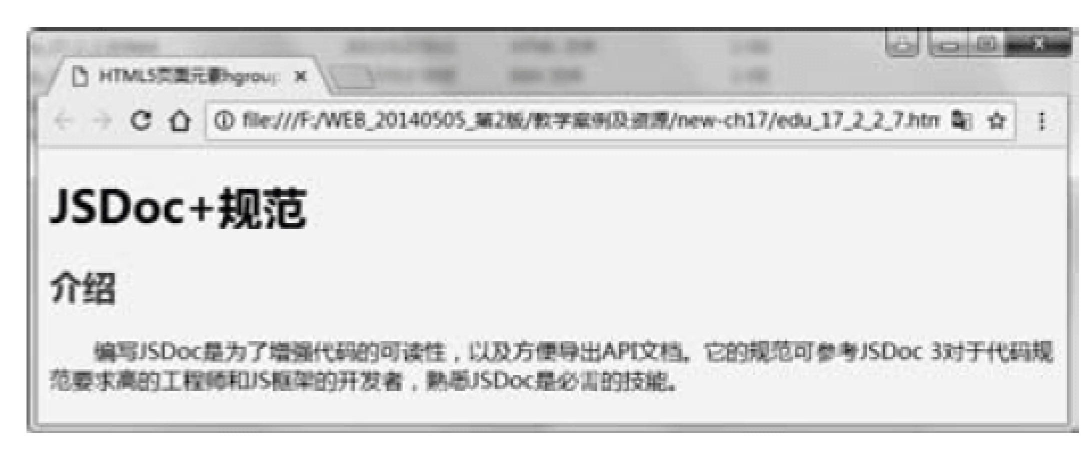



图 13-10　HTML5 页面元素 hgroup 应用

### 13.3.2 figure 标记与 figcaption 标记

figure 标记用于对元素进行组合，常用于图像与图像描述组合。figcaption（图题）标记用于定义 figure 元素的标题（caption），可以给一组图像标记定义标题，但 figcaption 标记不是必需的。如果包含了 figcaption 元素，那么它必须放置在 figure 元素的第一个或最后一个子元素的位置上。

【例 13-3-2】figure 与 figcaption 标记的应用。代码如下，页面效果如图 13-11 所示。

视频讲解

```
<!-- edu_13_3_2.html -->
<!doctype html>
<html lang="en">
    <head>
        <meta charset="UTF-8">
        <title>HTML5页面元素figure与figcaption标记的应用</title>
    </head>
    <body>
        <figure>
```

```
            <p>HTML5具有语义、离线与存储、设备访问等八个新特性，其对应的logo如下图所示：</p>
            <img src="class-header-semantics.jpg" width="150px" alt="语义" title="语义" />
            <img src="class-header-offline.jpg" width="150px" alt="离线&存储" title="离线&存储" />
            <img src="class-header-device.jpg"  width="150px" alt="设备访问" title="设备访问" />
            <figcaption >HTML5新logo(图题)</figcaption>
        </figure>
    </body>
</html>
```




图 13-11　HTML5 页面元素 figure 与 figcaption 应用

## 13.3.3　mark 标记与 time 标记

记号 mark 标记用来定义带有记号的文本。在需要突出显示文本时可以使用 mark 标记。此标记对关键字做高亮处理（黄底色标注），突出显示，标注重点，在搜索方面可以应用。

时间 time 标记用来定义公历的时间（24 小时制）或日期，时间和时区偏移是可选的。该标记能够以机器可读的方式对日期和时间进行编码。该标记不会在任何浏览器中呈现任何特殊效果。

### 1. 基本语法

```
<mark>重点标注的内容</mark>
<time>9:00</time>   <!-- 定义时间 -->
<time datetime="2017-05-01" pubdate="pubdate">国际劳动节</time> <!—定义日期 -->
```

### 2. 属性说明

time 标记的 pubdate 属性：指示该标记中的日期/时间是文档（或最近的 article 标记）的发布日期。

time 标记的 datetime 属性：规定日期/时间。否则，由元素的内容给定日期/时间。

**【例 13-3-3】** mark 和 time 标记的应用。代码如下，页面效果如图 13-12 所示。

视频讲解

```
<!-- edu_13_3_3.html -->
<!doctype html>
<html lang="en">
    <head>
        <meta charset="UTF-8">
        <title>HTML5页面元素mark和time标记的应用</title>
        <script type="text/javascript" src="html5shiv.js"></script>
    </head>
    <body>
```

```
        <article>
            <header>
                <h1>五一国际劳动节</h1>
            </header>
            <p style="text-indent:2em;">国际劳动节又称"<mark>五一国际劳动节
</mark>"、"<mark>国际示威游行日</mark>"(International Workers' Day或者May Day),
是世界上80多个国家的全国性节日。定在每年的五月一日。它是全世界劳动人民共同拥有的节日。1889
年7月，由恩格斯领导的第二国际在巴黎举行代表大会。会议通过决议，规定<time
datetime="1890-05-01">1890-05-01</time>国际劳动者举行游行，并决定把5月1日这一天定为
国际劳动节。中央人民政府政务院于1949年12月作出决定，将5月1日确定为劳动节。1989年后，国务
院基本上每5年表彰一次全国劳动模范和先进工作者，每次表彰3000人左右。</p>
        </article>
    </body>
</html>
```

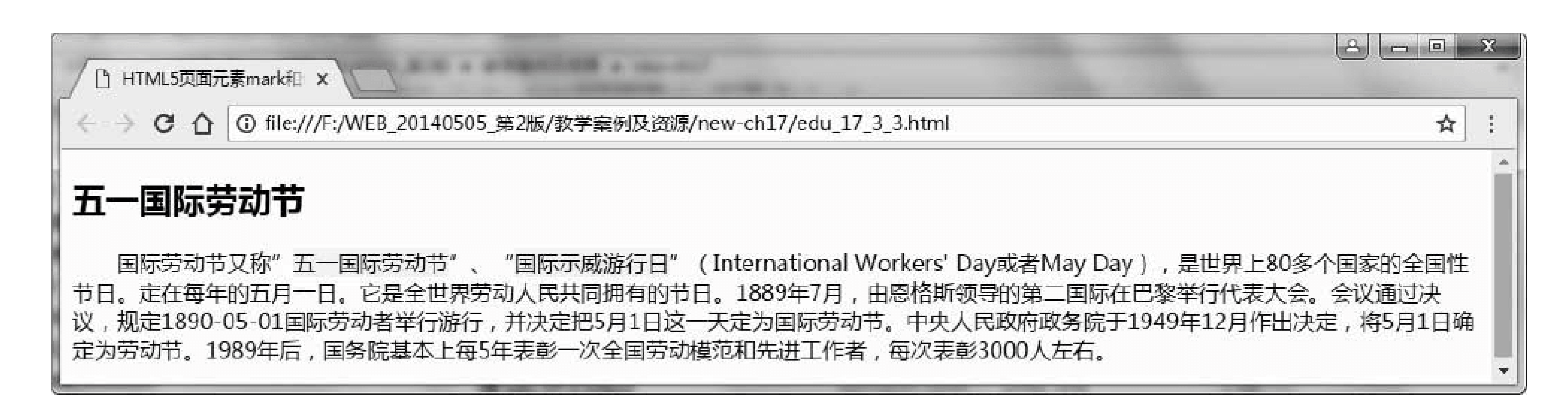



图 13-12　HTML5 页面元素 mark 与 time 应用

## 13.3.4　details 标记与 summary 标记

细节 details 标记是一个开关式、交互式控件，用来定义用户可见的或者隐藏的需求补充细节，任何形式的内容都能被放在该标记中。该元素的内容对用户是不可见的，除非设置了 open 属性。与摘要 summary 标记配合使用可以为 details 定义标题，summary 元素应该是 details 元素的第一个子元素。标题是可见的，用户单击标题时，会显示出 details。只有 Chrome、Safari 6 以上支持 summary 标记。

**基本语法**

```
<details open>
    <summary> details的标题</summary>
    details的详细内容
</details>
```

**【例 13-3-4】** details 和 summary 标记的应用。代码如下，页面效果如图 13-13 所示。

视频讲解

```
<!-- edu_13_3_4.html -->
<!doctype html>
<html lang="en">
    <head>
        <meta charset="UTF-8">
        <title>HTML5页面元素details和summary标记的应用</title>
        <script type="text/javascript" src="html5shiv.js"></script>
    </head>
    <body>
        <details>
            <summary>HTML5是下一代的HTML。</summary>
            <h3>什么是HTML5？</h3>
            <p>HTML5将成为HTML、XHTML以及HTML DOM的新标准。</p>
```

```
            <p>HTML的上一个版本诞生于1999年。自从那以后，Web世界已经经历了巨变。</p>
            <p>大部分现代浏览器已经具备了某些HTML5支持。</p>
        </details>
        <p><strong><mark>注意：</mark></strong>目前，只有Chrome和Safari 6支持details标签。</p>
    </body>
</html>
```






图 13-13　HTML5 页面元素 details 与 summary 应用

### 13.3.5　progress 标记与 meter 标记

进度 progress 标记用来定义运行中的任务进度（进程）。该标记有两个属性：max 表示规定需要完成的值；value 规定进程的当前值。

度量 meter 标记定义已知范围或分数值内的标量测量，也被称为 gauge（尺度）。如磁盘用量、CPU 使用率等，meter 标记属性如表 13-4 所示。

**表 13-4　meter 标记的属性、值及说明**

| 属　　性 | 值 | 说　　明 |
| --- | --- | --- |
| form | form_id | 规定 meter 元素所属的表单 |
| high | number | 规定被界定为高值的范围 |
| low | number | 规定被界定为低值的范围 |
| max | number | 规定范围的最大值 |
| min | number | 规定范围的最小值 |
| optimum | number | 规定度量的最优值 |
| value | number | 必需。规定度量的当前值 |

**【例 13-3-5】** progress 和 meter 标记的应用。代码如下，页面效果如图 13-14 所示。

视频讲解

```
<!-- edu_13_3_5.html -->
<!doctype html>
<html lang="en">
    <head>
        <meta charset="UTF-8">
        <title>HTML5页面元素progress和meter标记的应用</title>
        <script type="text/javascript" src="html5shiv.js"></script>
    </head>
    <body>
     <p><strong>文件下载进度：</strong>
     <progress value="22" max="100">设置属性</progress></p>
     <p><strong>空进度条：</strong><progress>未设置属性</progress></p>
     <p><strong>服务器CPU使用情况：</strong>
     <meter value="0.3" high="0.9" low="0.1" optimum="0.5">3/10</meter> </p>
```

```
        <p><strong>内存使用情况：</strong><meter value="0.6" max="1" min="0" optimum=".75" >60%</meter></p>
        <p><mark>注释：</mark>IE9以及更早的版本不支持progress、meter 标记。</p>
    </body>
</html>
```




图 13-14　HTML5 页面元素 progress 与 meter 应用

IE9 或者更早版本的 IE 浏览器不支持 progress、meter 标记。在需要显示工作任务的进度时，通常将 JavaScript 脚本与 progress 标记结合起来使用。

### 13.3.6　input 标记与 datalist 标记

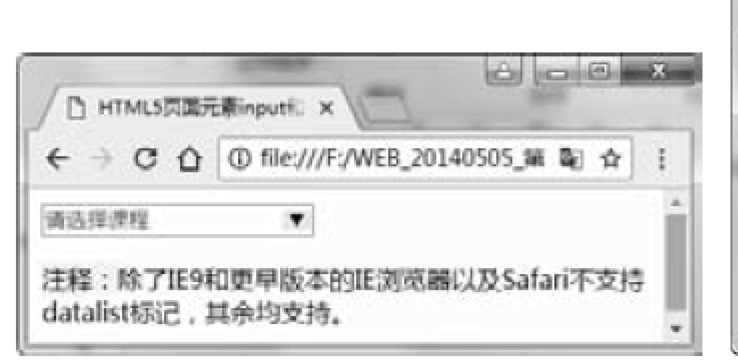
input 标记用于搜集用户信息。详细介绍请参见第 12 章，此处仅介绍通过 input 标记的 list 属性与 datalist 标记的 id 属性进行关联，即将此两个属性的值设置为相同的值，通过 datalist 标记列出所有合法的输入值列表。

选项列表 datalist 标记用来定义 input 标记可能的选项列表。一般与 input 标记配合使用，主要用来定义 input 可能的值，提供“自动完成”的功能，方便用户输入。datalist 标记及其选项不会被显示出来，只有当用户鼠标盘旋在 input 标记域时，才能看到“▼”，然后单击“▼”弹出一个下拉列表，提供用户选择作为用户的输入数据。

**【例 13-3-6】**input 和 datalist 标记的应用。代码如下，页面效果如图 13-15 所示。

视频讲解

```
<!-- edu_13_3_6.html -->
<!doctype html>
<html lang="en">
    <head>
        <meta charset="UTF-8">
        <title>HTML5页面元素input和datalist标记的应用</title>
        <script type="text/javascript" src="html5shiv.min.js"></script>
    </head>
    <body>
        <input list="courese" placeholder="请选择课程" />
        <datalist id="courese">
            <option value="HTML5移动应用开发">
            <option value=".NET应用开发">
            <option value="JavaEE应用开发">
            <option value="PHP+MySQL应用开发">
        </datalist>
         <p><mark>注释：</mark>除了IE9和更早版本的IE浏览器以及Safari不支持datalist标记，其余均支持。</p>
    </body>
</html>
```





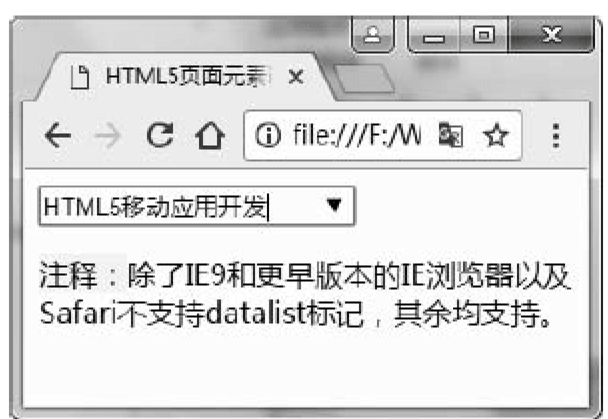



图 13-15　HTML5 页面元素 input 与 datalist 应用

除了 IE9 和更早版本的 IE 浏览器以及 Safari 不支持 datalist 标记，其余均支持。

还有其他一些页面元素，如 menu、ruby、rt、rp、output 等，可以参见表 13-2，此处不再一一介绍。

# 13.4　HTML5 表单

表单是 HTML 中获取用户输入的手段，HTML5 对表单系统做了彻底的改造，以适应当前的应用。在 HTML5 中增加了从用户收集特定类型数据的新方法和在浏览器中检查数据的能力，但在使用有些新增特性前最好先检查一下浏览器的支持情况。下面从表单新增属性、表单新增元素及表单新增类型等方面分别进行介绍。

## 13.4.1　HTML5 新增的表单属性

HTML5 给 form 标记新增了一些属性。这些属性是 autocomplete、novalidate。

**1. form 标记的新属性**

1）autocomplete 属性

autocomplete：on|off。属性规定 form 标记或类型为 text、search、url、telephone、email、password、date pickers、range、color 的 input 标记是否具有自动完成的功能。当表单元素设置了自动完成功能后，会记录用户输入过的内容，双击表单元素会显示历史输入。

2）novalidate 属性

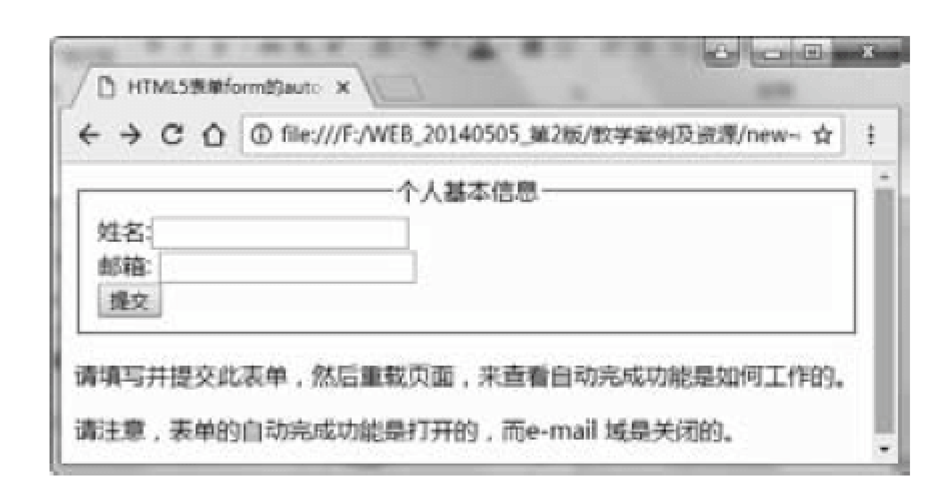
novalidate：true|false。属性规定在提交表单时不进行验证 form 或类型为 text、search、url、telephone、email、password、date pickers、range、color 的 input 标记。

**【例 13-4-1】**表单属性 autocomplete 和 novalidate 的应用。代码如下，页面效果如图 13-16 所示。赋值方法：novalidate="novalidate"或 novalidate="true"。

视频讲解

```
<!-- edu_13_4_1.html -->
<!doctype html>
<html lang="en">
    <head>
        <meta charset="UTF-8">
        <title>HTML5表单form的autocomplete和novalidate属性的应用</title>
        <script type="text/javascript" src="html5shiv.min.js"></script>
        </head>
    <body>
         <form action="" method="get" novalidate="novalidate" autocomplete="on">
             <fieldset>
              <legend align="center">个人基本信息</legend>
               姓名:<input type="text" name="name" /><br/>
```

```
                邮箱: <input type="email" name="email" autocomplete="off" /><br/>
               <input type="submit" value="提交" />
             </fieldset>
         </form>
         <p>请填写并提交此表单，然后重载页面，来查看自动完成功能是如何工作的。</p>
         <p>请注意，表单的自动完成功能是打开的，而e-mail 域是关闭的。</p>
     </body>
</html>
```



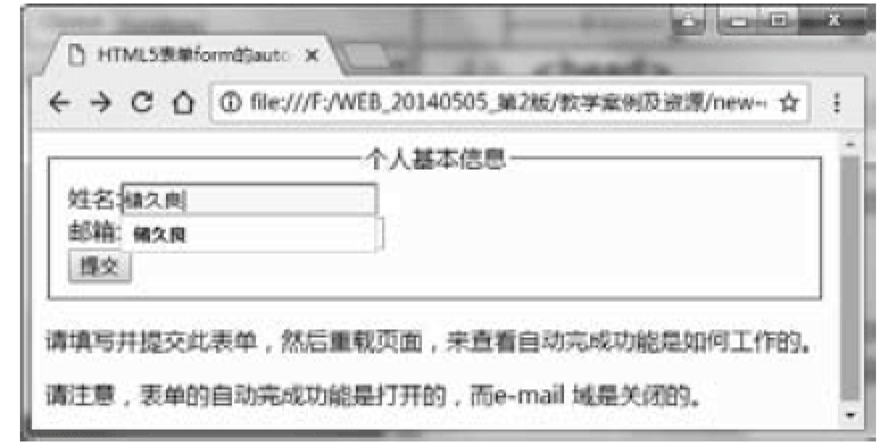



图 13-16　HTML5 表单 form 的 autocomplete 与 novalidate 属性的应用

**2．input 标记的新属性**

HTML5 给 input 标记新增一些属性。这些属性是 autocomplete、autofocus、form、form overrides(formaction、formenctype、formmethod、formnovalidate、formtarget)、height、width、list、min、max、step、multiple、pattern(regexp)、placeholder、required。

- height 和 width 属性。

height 和 width 属性规定只适用于 image 类型的 input 标记的图像高度和宽度。

- form 属性。

form 属性规定输入域所属的一个或多个表单。form 属性必须引用所属表单的 id。

- list 属性。

list 属性规定输入域的 datalist。datalist 标记是输入域的选项列表。list 属性适用于类型为 text、search、url、telephone、email、password、date pickers、range、color 的 input 标记。

- placeholder 属性。

placeholder 属性提供一种提示，描述输入域所期待的值。当用户将鼠标盘旋在该域上时，单击“▼”会弹出下拉列表选项，简短的提示在用户输入值前会显示在输入域上。方便用户快速选择输入。该属性支持类型为 text、search、url、telephone、email、password 的 input 标记。

- autofocus 属性。

autofocus 属性规定在页面加载时，该域自动地获得焦点。该属性适用于所有 input 标记的类型。属性设置方法为 autofocus="autofocus"或直接使用该属性 autofocus。

**【例 13-4-2】**input 标记的新增部分属性的应用。代码如下，页面效果如图 13-17 所示。

视频讲解

```
<!-- edu_13_4_2.html -->
<!doctype html>
<html lang="en">
   <head>
       <meta charset="UTF-8">
       <title>HTML5的input标记新增部分属性的应用</title>
       <script type="text/javascript" src="html5shiv.js"></script>
    </head>
```

```
<body>
    <fieldset style="text-align:center;border:1px solid red;">
        <legend align="center">用户登录</legend>
        <form name="myform" action="" method="get">
            姓名: <input type="text" name="name" placeholder="请输入姓名" autofocus="autofocus" />
            班级: <input type="text" name="class" placeholder="请输入班级" list="class_list"/>
            <datalist id="class_list">
                <option value="15计算机1班">
                <option value="15软件工程">
                <option value="15信息管理与信息系统">
                <option value="15电子信息工程">
            </datalist>
            <input type="image" src="eg_submit.jpg" width="35" height="35"/>
        </form>
        <p>下面的输入域在 form 元素之外，但仍然是表单的一部分。</p>
        密码: <input type="password" name="user_key" form="myform">
    </fieldset>
  </body>
</html>
```




图 13-17　input 标记的新增部分的应用

- required 属性。

required 属性规定必须在提交之前填写输入域（不能为空）。required 属性适用于类型为 text、search、url、telephone、email、password、date pickers、number、checkbox、radio、file 的 input 标记。属性设置方法为 required="required"或直接使用该属性 required。

- min、max 和 step 属性。

min、max 和 step 属性用于为包含数字或日期的 input 类型规定限定（约束）。其中 max 属性规定输入域所允许的最大值。min 属性规定输入域所允许的最小值。step 属性为输入域规定合法的数字间隔，例如 step="5"，则合法的数是 -5、0、5、10 等。该组属性适用类型为 date pickers、number、range 的 input 标记。

- multiple 属性。

multiple 属性规定输入域中可选择多个值。适用于类型为 email、file 的 input 标记。

- form overrides 表单重写属性。

表单重写属性（form override attributes）允许重写 form 元素的某些属性设定。这些重写属性分别是重写表单的 action 属性 formaction、重写表单的 enctype 属性 formenctype、重写表单的 method 属性 formmethod、重写表单的 novalidate 属性 formnovalidate、重写表单的 target 属性 formtarget。表单重写属性适用于类型为 submit 和 image 的 input 标记。

- pattern 属性。

pattern 属性规定用于验证 input 域的模式（pattern）。pattern 属性适用于类型为 text、search、url、tel、email、password 的 input 标记。该属性值是正则表达式。

【例 13-4-3】input 标记的新增其他属性的应用。代码如下，页面效果如图 13-18～图 13-22 所示。

视频讲解

```
<!-- edu_13_4_3.html -->
<!doctype html>
<html lang="en">
    <head>
        <meta charset="UTF-8">
        <title>HTML5的input标记新增部分属性的应用</title>
        <script type="text/javascript" src="html5shiv.js"></script>
    </head>
    <body>
        <form action="" method="get">
            <fieldset style="text-align:center;padding:20px;">
             <legend align="center">理财认购信息</legend>
              用户名称<input type="text" name="usrname" required><!-- 不能为空 -->
             认购金额：<input type="number" name="money" min="5" max="100" step="5"/>
                 手机号码：<input type="text" name="phone"  pattern="1[3|4|5|8][0-9][0-9]{8}$" title="手机号码是11位数字" required /><br/><br/><!-- 不能为空且必须为11位数字 -->
        <label>相片：</label><input type="file" multiple="multiple"/><!-- 支持多选 -->
              <input type="submit" formaction="admin.asp" value="以管理员身份提交" /><!-- 重写action -->
               <input type="submit" formnovalidate="true" value="不要求验证提交" /><!-- 重写novalidate -->
              <input type="submit" value="提交" />
            </fieldset>
        </form>
    </body>
</html>
```




图 13-18　新增表单其他属性应用初始页面

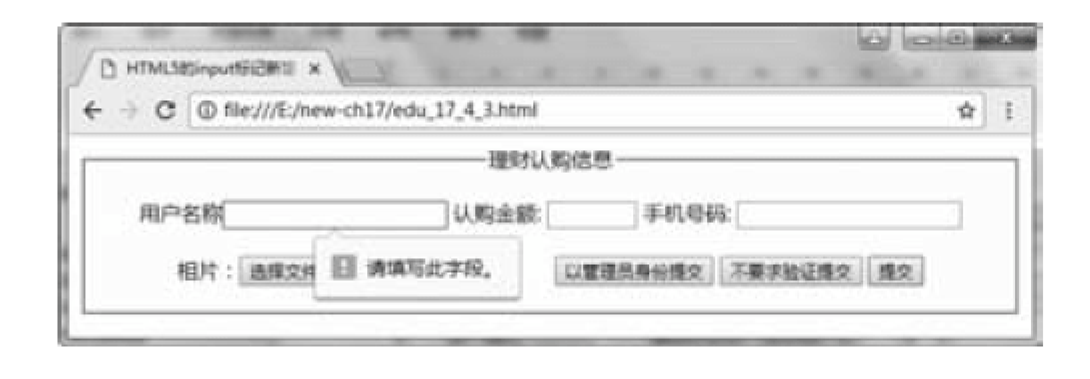



图 13-19　未输入用户名称直接提交后的页面

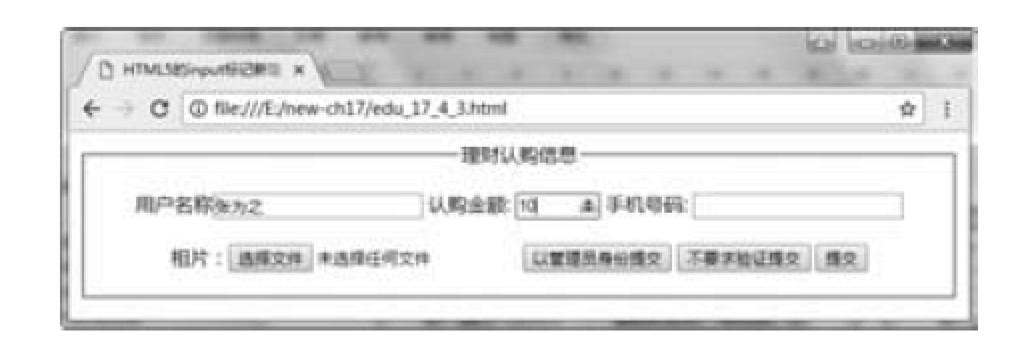



图 13-20　number 型 input 标记的属性应用

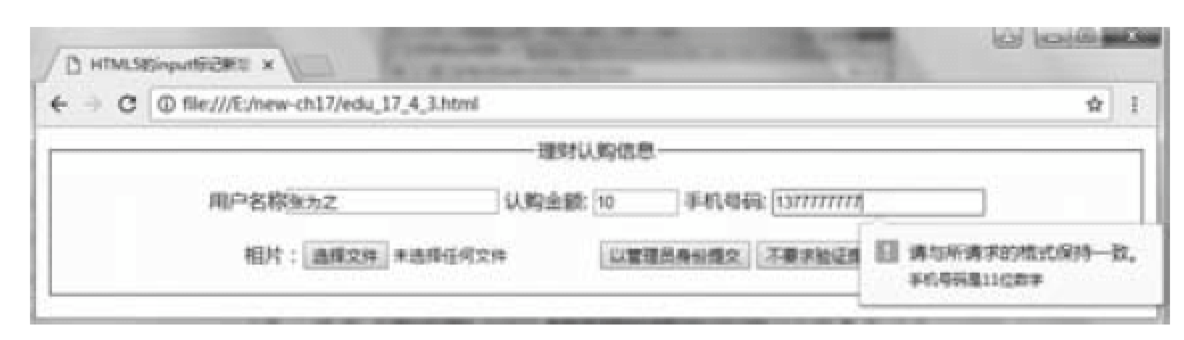



图 13-21　设置 pattern 属性后验证信息




图 13-22　给 file 类型的 input 标记设置 multiple 属性支持多选

## 13.4.2　HTML5 新增的表单元素

HTML5 新增 output、keygen、datalist 等表单元素，其功能描述如表 13-5 所示。

**表 13-5　HTML5 新增表单元素**

| 标 记 名 称 | 标记功能描述 |
| --- | --- |
| <output></output> | 定义不同类型的输出，例如脚本的输出 |
| <keygen></keygen> | 规定用于表单的密钥对生成器字段 |
| <datalist></datalist> | 定义选项列表。与 input 元素配合使用该元素，来定义 input 可能的值 |

### 1. output 标记

output 标记定义不同类型的输出。该标记有 for、form、name 三个属性。for 属性用于描述计算中使用的元素与计算结果之间的关系，其值为每一元素的 id，多个 id 之间用空格分隔。form 属性用于定义输入字段所属的一个或多个表单。name 属性用于定义对象的唯一名称（表单提交时使用）。

**【例 13-4-4】** 新增 output 标记的应用。代码如下，页面效果如图 13-23 所示。

视频讲解

```
<!-- edu_13_4_4.html -->
<!doctype html>
<html lang="en">
   <head>
      <meta charset="UTF-8">
      <title>HTML5的input标记新增部分属性的应用</title>
      <script type="text/javascript" src="html5shiv.js"></script>
   </head>
   <body>
      <form oninput="sum.value=parseInt(num1.value)+parseInt(num2.value)">
         0<input type="range" id="num1" value="50" min="0" max="100">100
         +<input type="number" id="num2" value="50">
        =<output name="sum" for="num1 num2"></output>
      </form>
      <p><strong>注意:</strong>IE浏览器不支持output标记。</p>
   </body>
</html>
```

代码中 range 类型的 input 标记的表示范围为 0～100，当前值为 50。number 类型的 input 标记的当前值为 50。当用户拖动滑块时，右边的 output 标记内容通过 oninput 事件句柄绑定 JS 语句 sum.value=parseInt(num1.value)+parseInt(num2.value)来自动计算并填充。




图 13-23　output 标记的应用(左：初始时，右：拖动滑块时)

**2．keygen 标记**

keygen 标记用来提供一种验证用户的可靠方法。keygen 元素是密钥对生成器（key-pair generator）。当提交表单时，会生成两个键：一个是私钥（private key），一个公钥（public key）。私钥存储于客户端，公钥则被发送到服务器。公钥可用于之后验证用户的客户端证书（client certificate）。目前，浏览器对此元素的糟糕的支持度不足以使其成为一种有用的安全标准。

【例 13-4-5】新增 keygen 标记的应用。代码如下，页面效果如图 13-24 所示。

```
<!-- edu_13_4_5.html -->
<!DOCTYPE html>
<html>
    <head>
        <meta charset="UTF-8">
        <title>keygen标记的应用</title>
    </head>
    <body>
        <form action="" method="get">
        用户名: <input type="text" name="usr_name" />
        加密: <keygen name="security" />
          <input type="submit" value="提交" />
     </form>
    </body>
</html>
```

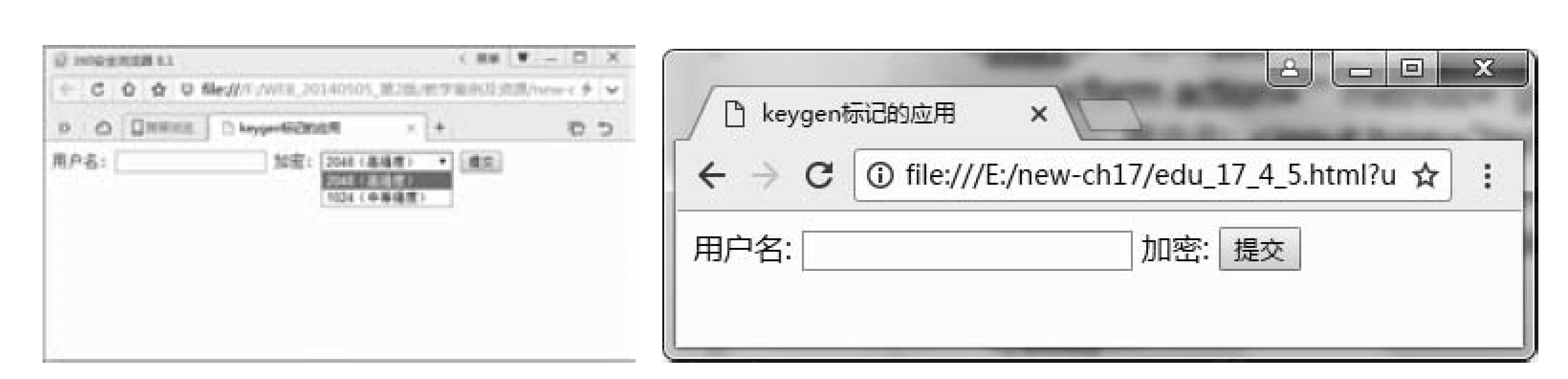



图 13-24　keygen 标记的应用(左：支持，右：不支持)

**3．datalist 标记**

datalist 标记规定了 input 标记可能的选项列表。datalist 标记被用来为 input 标记提供“自动完成”的特性。用户能看到一个下拉列表，里边的选项是预先定义好的，将作为用户的输入数据。一般使用 input 标记的 list 属性来绑定 datalist 元素（input 标记的 list 属性值应该与 datalist 标记的 id 属性值相同）。案例参见例 13-3-6 和例 13-4-2。

## 13.4.3　HTML5 新增的 input 类型

HTML5 增加很多新的表单输入类型。这些新特性提供了更好的输入控制和验证。新增 input 标记的输入类型是 color、date pickers（日期选择器，包括 date、month、week、time、datetime、datetime-local）、email、number、range、search、tel、url。目前所有的主流浏览

器一般都支持新的 input 类型，即使不被支持，仍可以显示为常规的文本域。以下对这些新增类型的使用方法分别进行介绍。

- Input 类型——Date Pickers（日期选择器）。

HTML5 提供多个可供选取日期和时间的新输入类型：

（1）date——选取日、月、年。

（2）month——选取月、年。

（3）week——选取周和年。

（4）time——选取时间（小时和分钟）。

（5）datetime——选取时间、日、月、年（UTC 时间）。

（6）datetime-local——选取时间、日、月、年（本地时间）。

**【例 13-4-6】**表单日期选择器的应用。代码如下，页面效果如图 13-25 所示。

视频讲解

```
<!-- edu_13_4_6.html -->
<!DOCTYPE html>
<html>
    <head>
        <meta charset="UTF-8">
        <title>表单新增input类型的应用</title>
    </head>
    <body>
        <fieldset>
            <legend align="center">新生报到须知</legend>
            开学日期: <input type="date" /><br />
            开学起始周: <input type="week" name="user_date" /><br />
            开始起始月: <input type="month" name="user_date" /><br />
            交费时间: <input type="time" name="user_date" /><br />
            日期与时间: <input type="datetime" name="user_date" /><br />
            本地日期与时间: <input type="datetime-local" name="user_date" /><br />
            <input type="submit" value="提交" />
            <input type="reset" />
            </fieldset>
    </body>
</html>
```

当鼠标盘旋在 datetime 类型的文本框时，不会出现任何按钮。当鼠标盘旋在 time 类型的文本框上时，会出现微调按钮。当鼠标盘旋在其余类型均文本框时，会出现组合按钮“ ▾▼”（微调+下三角），单击“▼”按钮以弹出日期选择器，进行相关选择设置；也可以利用上、下微调按钮分别进行年、月、日、时、分等单项设置。




图 13-25　input 标记的日期选择器的应用（左：初始盘旋时，右：设置完成时）

- input 类型：color。

```
<input type="color" name="favcolor"> <!-- 从取色器拾取颜色 -->
```

- input 类型：tel。

定义输入电话号码字段。

```
<input type="tel" name="usrtel">
```

- input 类型：email。

email 类型用于包含 e-mail 地址的输入域。在提交表单时，会自动验证 email 域的值是否合法有效。

```
<input type="email" name="useremail"> <!-- 自动验证邮箱格式 -->
```

- input 类型：number。

number 类型用于包含数值的输入域。此类型的 input 标记常用属性如表 13-6 所示。

```
<input type="number" name="mynumber" min="0" max="100">
```

**表 13-6　number 类型 input 标记的属性及说明**

| 属　　性 | 说　　明 |
|---|---|
| disabled | 规定输入字段是禁用的 |
| max | 规定允许的最大值 |
| maxlength | 规定输入字段的最大字符长度 |
| min | 规定允许的最小值 |
| pattern | 规定用于验证输入字段的模式 |
| readonly | 规定输入字段的值无法修改（只读） |
| required | 规定输入字段的值是必需的 |
| size | 规定输入字段中的可见字符数 |
| step | 规定输入字段的合法数字间隔 |
| value | 规定输入字段的默认值 |

- input 类型：range。

range 类型用于包含一定范围内数字值的输入域。range 类型显示为滑动条。

```
<input type="range" name="money" min="1" max="1000" step="5">
```

- input 类型：search。

search 类型用于搜索域，例如站点搜索或 Google 搜索。

```
<input type="search" name="websidesearch">
```

- input 类型：url。

url 类型用于包含 URL 地址的输入域。在提交表单时，会自动验证 url 域的值。

```
<input type="url" name="homepage">
```


视频讲解

**【例 13-4-7】**其他新增 input 类型综合应用。代码如下，页面效果如图 13-26 和图 13-27 所示。



图 13-26　新增其他 input 类型初始页面



图 13-27　“颜色”对话框界面

```
<!-- edu_13_4_7.html -->
<!DOCTYPE html>
<html>
    <head>
        <meta charset="UTF-8">
        <title>表单新增input类型的应用</title>
        <script type="text/javascript" src="html5shiv.min.js"></script>
    </head>
    <body>
        <fieldset style="width:500px;height:200px;padding:20px 50px;">
            <legend align="center">新增其他input类型</legend>
            <form method="post" action="">
                设置颜色：<input type="text" name="color1" id="color1" readonly>
                <input type="color" name="color2" oninput="color1. value=color2.value"><br>
                输入邮箱：<input type="email" name="useremail"><br>
                站内搜索：<input type="search" name="insidesearch"><br>
                电话号码：<input type="tel" name="usrtel"><br>
                个人主页：<input type="url" name="homepage"><br>
                年龄：<input type="range" name="age" min="1" max="120" oninput="age_num.value=age.value"><output name="age_num" for="age"></output><br>
                 期望薪酬：<input type="number" name="quantity" min="2500" max="10000" step="100" value="2500"><br>
        <input type="submit" value="提交" />
        <input type="reset" />
        </form>
        </fieldset>
    </body>
</html>
```

在图 13-26 中单击 color 类型文本域，弹出“颜色”对话框，如图 13-27 所示。单击某一颜色区域后，再单击“确定”按钮，将六位十六进制颜色填充到左边的文本框中。由于定义邮箱为 email 类型，会自动验证，所以当用户输入的邮箱不包含@、.等字符时会弹出验证信息，如图 13-28 所示。当用户在个人主页对应的 url 类型的文本域中输入信息不正确时，会弹出验证信息，如图 13-29 所示。



图 13-28　email 类型文本域验证页面



图 13-29　url 类型文本域验证页面

当用户拖动年龄滑动条时，会将滑动条的当前值填充到右边的 output 标记内。如图 13-30 所示。当用户设置期望薪酬时，通过单击微调按钮来递增或递减薪酬标准，设置 min 为 2500，max 为 10000，step 为 100，所以此域中初始值为 2500，单击微调按钮每次自动递增或递减 100，如图 13-31 所示。



图 13-30　拖动年龄滑动条时的页面



图 13-31　单击微调按钮时的页面

# 13.5　HTML5 视频与音频

大多数商业网站都喜欢采用视频来宣传自己的公司或推销自己的产品或服务。然而在 HTML4.01 版本基础之前，只能通过相关插件（比如 Flash）来播放，而且所有浏览器不一定都有同样的插件，还需要安装其他插件才能实现。HTML5 提供了 video 标记和 audio 标记，很好地解决这一问题。

## 13.5.1　video 标记及属性

HTML5 规定了一种通过 video 元素来包含视频的标准方法。Video 标记支持三种视频格式，分别为 MP4、WebM、Ogg。其格式的说明如下：

- Ogg：带有 Theora 视频编码和 Vorbis 音频编码的 Ogg 文件。
- MPEG4：带有 H.264 视频编码和 AAC 音频编码的 MPEG 4 文件。
- WebM：带有 VP8 视频编码和 Vorbis 音频编码的 WebM 文件。
- Video 标记提供了播放、暂停和音量控件来控制视频。

**1. 基本语法**

```
<video src="movie.ogg" width="320" height="240" controls="controls">
    您的浏览器不支持 video标记。
</video>
```

**2. 属性说明**

width 和 height 属性：控制视频的尺寸。使用时需要设置视频的高度和宽度，便于视频播放。如果不设置宽度和高度，页面就会根据原始视频的大小而改变。

src 属性：规定要播放的视频的 url。

controls 属性：设置该属性，则页面上会显示播放控件。浏览器控件应该包括播放、暂停、定位、音量、全屏切换、字幕（如果可用）、音轨（如果可用）。

autoplay 属性：设置该属性，则视频就绪后马上播放。设置方法：autoplay="autoplay " 或 autoplay。

loop：设置该属性，则当媒体文件完成播放后再次开始播放。

preload：设置该属性，则视频在页面加载时进行加载，并预备播放。如果使用 autoplay，则忽略该属性。该属性有三种值：auto（一旦页面加载，则开始加载音频/视频）、metadata（当页面加载后仅加载音频/视频的元数据）、none（页面加载后不应加载音频/视频）。格式如下：

```
<video preload="auto|metadata|none">
```

poster 属性：用于在视频下载时显示的图像（海报图片），或者在用户点击播放按钮前显示的图像。如果未设置该属性，则使用视频的第一帧来代替。赋值方法：poster="url"。

如果浏览器不支持 video 标记，就在<video>与</video>标记之间插入相关提示信息。

video 标记支持多个 source 标记。可以使用 source 标记为 video 标记和 audio 标记提供多个不同的音频、视频文件，以解决浏览器支持。如果浏览器支持将使用第一个可识别的格式。IE8 或者更早的 IE 版本不支持 video 标记。使用语法如下所示。

```
<video width="320" height="240" controls="controls">
  <source src="movie.ogg" type="video/ogg">
  <source src="movie.mp4" type="video/mp4">
  您的浏览器不支持 video标记。
</video>
```

**【例 13-5-1】** video 标记的应用。代码如下，页面效果如图 13-32 和图 13-33 所示。

视频讲解

```
<!--   edu_13_5_1.html-->
<!DOCTYPE html>
<html>
   <head>
      <meta charset="UTF-8">
      <title>视频标记的应用</title>
   </head>
   <body>
      <fieldset style="text-align:center;float:left;">
         <legend>src属性提供视频文件</legend>
         <video src="movie.ogg"  poster="url" loop autoplay width="320" height="240" controls="controls">
             您的浏览器不支持 video标记。
         </video>
      </fieldset>
      <fieldset style="text-align:center;float:left;">
         <legend>source标记提供不同的视频文件</legend>
         <video width="320" height="240" controls="controls">
         <source src="movie.ogg" type="video/ogg">
         <source src="movie.mp4" type="video/mp4">
             您的浏览器不支持 video标记。
         </video>
      </fieldset>
   </body>
</html>
```





图 13-32　HTML5 视频播放页面




图 13-33　浏览器不支持视频时的页面

## 13.5.2　audio 标记及属性

HTML5 规定了一种通过 audio 元素来包含音频的标准方法。audio 标记能够播放声音文件或者音频流。同样可以使用 source 标记给 audio 标记提供不同格式的音频文件，浏览器将使用第一个支持的音频文件。audio 标记的部分属性与 video 标记相同，此处不再介绍。如果浏览器不支持 audio 标记，则显示 audio 标记之间的提示信息，如图 13-34 所示。IE8 及更早 IE 版本不支持 audio 标记。

**【例 13-5-2】**audio 标记的应用。代码如下，页面效果如图 13-34 和图 13-35 所示。

视频讲解

```
<!-- edu_13_5_2.html-->
<!DOCTYPE html>
<html>
    <head>
        <meta charset="UTF-8">
        <title>音频标记的应用</title>
        <script type="text/javascript" src="html5shiv.js"></script>
    </head>
    <body>
        <fieldset style="text-align:center;float:left;">
            <legend>src属性提供音频文件</legend>
            <audio src="horse.ogg" controls="controls">
                您的浏览器不支持 audio标记（元素）。
            </audio>
        </fieldset>
        <fieldset style="text-align:center;float:left;">
            <legend>source标记提供不同的音频文件</legend>
            <audio controls="controls">
                <source src="horse.ogg" type="audio/ogg">
                <source src="horse.mp3" type="audio/mpeg">
                您的浏览器不支持 audio标记（元素）。
            </audio>
        </fieldset>
    </body>
</html>
```

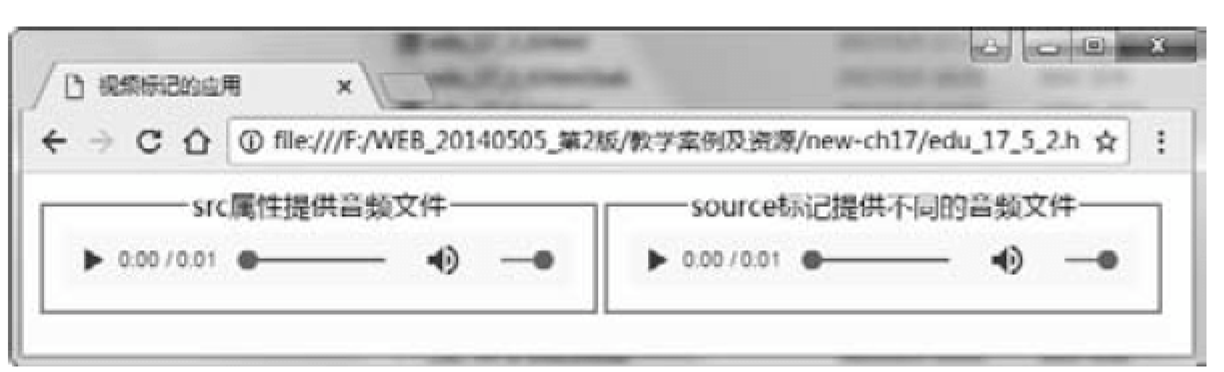



图 13-34　HTML5 音频播放页面

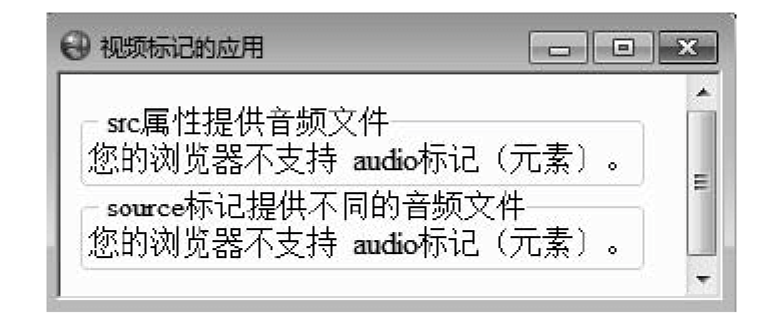



图 13-35　浏览器不支持音频时的页面

# 13.6　CSS3 基础应用

随着 Web 技术不断发展和广泛传播，原来的 CSS2 标准和相关技术似乎已经满足不了日益增长的开发需求。人们需要实现更加美观、用户体验更好的界面。CSS3 这个新一代的标准应运而生。

## 13.6.1　CSS3 新特性

为了满足 Web UI 的开发需求，它提供了一系列强大的功能，如许多新的 CSS 属性（文字、布局、颜色等）、各种 CSS 特效、CSS 动画、元素的变换等。这些 CSS 新特性可以说都是功能非常强大和比较完善的，用户只需要添加几行简单的 CSS 代码便可以实现出一系列令人眼前一亮的效果，相比之前用 JavaScript 去模拟这样的效果要好得多，不仅降低了复杂度，变得易维护，在性能上也突飞猛进了。

CSS3 被细分为许多“模块”。CSS2 中已经拆分成小块，又新增加了一些最重要的 CSS3 模块，分别为选择器、盒模型、背景和边框、文字特效、2D/3D 转换、动画、多列布局、用户界面。许多新的 CSS3 属性已在目前主流的浏览器中得到应用。本节主要详细介绍边框、转换、过渡与动画等 CSS3 新特性，以满足用户的学习与使用需要。

## 13.6.2　CSS3 浏览器兼容性

### 1．常用的浏览器属性前缀

为了让 CSS 样式能够满足不同浏览器版本的需要，充分发挥 CSS3 的魅力，需要在样式属性前面增加一些区分不同浏览器的前缀。

-webkit-：适用于 webkit 核心浏览器，包含 Safari、Chrome 等。

-moz-：适用于 Firefox 浏览器等。

-ms-：适用于 IE 浏览器。

-o-：适用于 Opera 浏览器。

在实际开发过程中，为了满足不同浏览器对 CSS3 新特性的支持，需要写上类似如下的声明语句。格式如下所示：

```
div{
    -moz-animation: myfirst 5s;     /* Firefox */
    -webkit-animation: myfirst 5s;  /* Safari 和 Chrome */
    -o-animation: myfirst 5s;       /* Opera */
    -ms-animation: myfirst 5s;      /* IE */
    animation: myfirst 5s;          /* 标准属性写在最后，*/
}
```

### 2．CSS3 前缀解决方案

为了使得 CSS 属性支持所有的浏览器，例如 Chrome、Firefox、IE、Opera、Safari 等，经常需要添加一大堆带有浏览器特定前缀的 CSS 相关代码，特别是 CSS3 中的相关属性，尤其需要处理。为了简化开发过程和相关的代码冗余问题，在页面中引入了-prefix-free 这个类库，可以自动帮助在 CSS 中添加相关的浏览器特有的前缀属性。

-prefix-free 是一个 JavaScript 工具库，用户再也不需要编写带有浏览器前缀的 CSS 代

码，在需要的时候，-prefix-free 会自动帮助添加当前浏览器需要的前缀。引用方式如下：

```
<script src="http://cdn.gbtags.com/prefixfree/1.0.7/prefixfree.min.js"></script>
<script src=”http://leaverou.github.com/prefixfree/prefixfree.min.js”></script>
```

-prefix-free 这个类库也可以从 http://leaverou.github.com/prefixfree/ 网站上直接下载到本地，使用下列格式来引用。

```
<script src="js/prefixfree.min.js"></script>
```

**3．CSS 样式重置方案**

由于不同的浏览器定义的 HTML 元素的默认样式不尽相同，导致页面在不同的浏览器中显示会有一定的差异，为了保护所有浏览器默认样式而不是完全去掉它们，推荐使用 Normalize.css 来统一不同浏览器下的样式。

Normalize.css 是一个很小的 CSS 文件，但它在默认的 HTML 元素样式上提供了跨浏览器的高度一致性。相比于传统的 CSS reset，Normalize.css 是一种现代的、为 HTML5 准备的优质替代方案。Normalize.css 现在已经被用于 Twitter Bootstrap、HTML5 Boilerplate、GOV.UK、Rdio、CSS Tricks 以及许许多多其他框架、工具和网站上。

用户可以从 Github 下载 Normalize.css，然后引用到页面中就可以。也可以在 Normalize.css 源码的基础上重新编写，在必要的时候用自己写的 CSS 覆盖默认值。

```
<link rel="stylesheet" href="css/normalize.css" type="text/css">
```

### 13.6.3 CSS3 边框

CSS3 之前，要想创建带圆角边框、添加阴影、绘制图形边框往往需要借助于类似 PhotoShop 这样的软件。有了 CSS3 后，一切问题迎刃而解。

CSS3 具有三个边框属性，如表 13-7 所示。

**表 13-7　CSS3 边框属性及说明**

| 属　　性 | 说　　明 |
|---|---|
| border-image | 设置所有 border-image-*属性的复合属性 |
| border-radius | 设置所有四个 border-*-radius 属性的复合属性 |
| box-shadow | 向矩形方框添加一个或多个阴影 |

**1．border-radius 圆角边框**

语法：

```
border-radius: 水平半径(1～4个值)px|%/垂直半径(1～4个值)px|%;
```

该属性是复合属性，用于设置四个 border-*-radius 属性。有两个参数，使用“/”分隔，第一个参数表示水平半径，可以有 1～4 个值；第二个参数表示垂直半径，也有 1～4 个值；若不使用“/”，说明水平与垂直半径相同。属性值可以是像素，也可以是%。每个半径的四个值的顺序是：左上角、右上角、右下角、左下角。如果省略左下角，右上角是相同的。如果省略右下角，左上角是相同的。如果省略右上角，左上角是相同的。圆角半径表示图如图 13-36 所示。

```
border-radius:2em;                    /* 等同于下列4行定义  */
border-top-left-radius:2em;           /* 定义左上角半径  */
border-top-right-radius:2em;          /*定义右上角半径*/
border-bottom-right-radius:2em;       /*定义右上角半径  */
border-bottom-left-radius:2em;        /*定义左下角半径  */
```

例如，设置 border-radius:10px 20px 30px 40px; 则说明四个圆角的水平与垂直半径各自相同，如图 13-37 左图所示。设置 border-radius:10px 20px 30px 40px/20px 50px 30px 10px; 则表示左上角、右上角、右下角、左下角的水平半径分别为 10px、20px、30px、40px；左上角、右上角、右下角、左下角的垂直半径分别为 20px、50px、30px、10px，如图 13-37 右图所示。

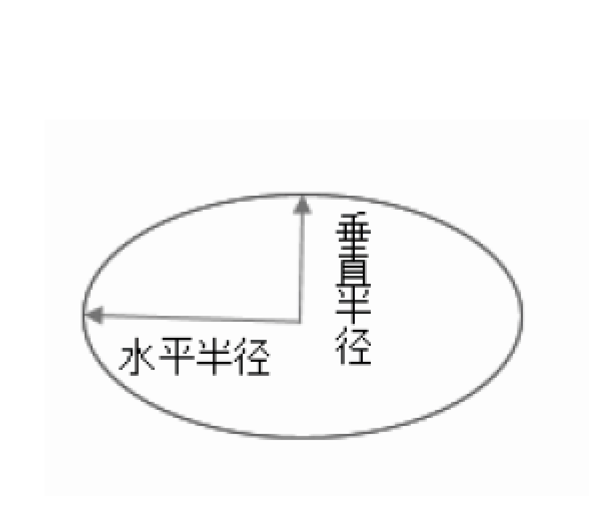



图 13-36　圆角的半径表示图

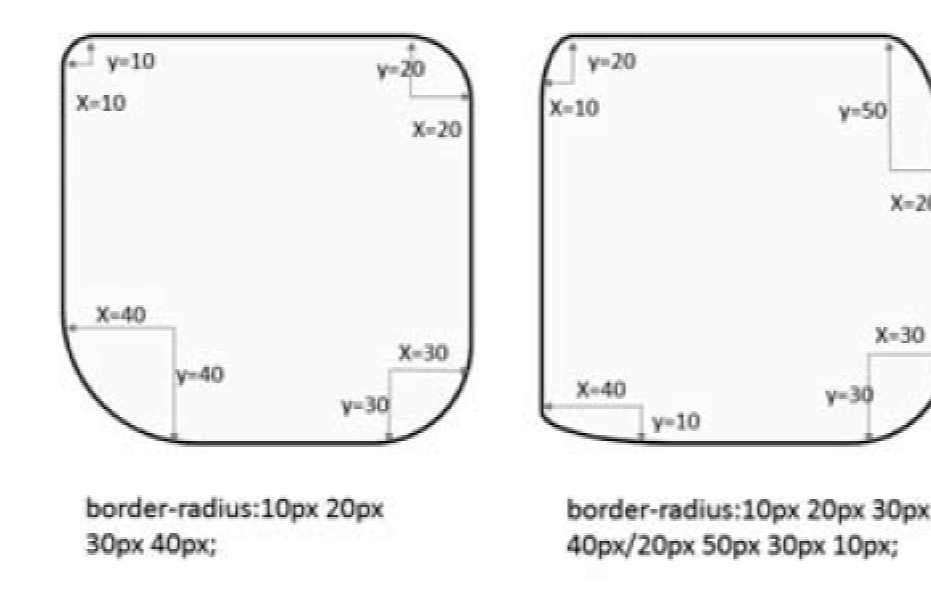



图 13-37　不同圆角的不同半径表示图

【例 13-6-1】CSS3 圆角边框的应用。代码如下，页面效果如图 13-38 所示。




图 13-38　CSS3 圆角边框的应用

视频讲解

```
<!-- edu_13_6_1.html -->
<!doctype html>
<html lang="en">
    <head>
        <meta charset="UTF-8">
        <title>CSS3边框的应用</title>
        <script type="text/javascript" src="html5shiv.js"></script>
        <link rel="stylesheet" href="css/normalize.css" type="text/css">
        <script type="text/javascript" src="js/prefixfree.min.js"></script>
        <style type="text/css">
            div{float:left;width:120px;height:120px;margin:50px 80px;
background:#dadada;border:6px solid #00cc66;padding:10px;    }
            #div1{    border-radius:25px;}
            #div2{    border-radius:25px 50px;}
            #div3{    border-radius:80px 100px 60px 120px/50px 60px 70px 70px;}
```

```
        </style>
    </head>
    <body>
        <h3>CSS3圆角边框</h3><hr>
        <div id="div1" class=""><p>半径均相同</p></div>
        <div id="div2" class=""><p>左、右对角的半径相同</p></div>
        <div id="div3" class=""><p>每个角水平与垂直半径不同</p>
        </div>
    </body>
</html>
```

**2．box-shadow 边框阴影**

box-shadow 边框阴影是复合属性，含有六个属性，如表 13-8 所示。

**表 13-8 box-shadow 属性值及说明**

| 值 | 说　明 |
|---|---|
| h-shadow | 必需。水平阴影的位置。允许负值 |
| v-shadow | 必需。垂直阴影的位置。允许负值 |
| blur | 可选。模糊距离 |
| spread | 可选。阴影的尺寸 |
| color | 可选。阴影的颜色。请参阅 CSS 颜色值 |
| inset | 可选。将外部阴影 (outset) 改为内部阴影 |

该属性是复合属性，可以设置六个属性值，分别表示水平偏移、垂直偏移、模糊距离、阴影尺寸、颜色、阴影模式（默认是外部阴影，指定 inset 改为内部阴影），其中阴影不占空间。

语法：

例如，box-shadow: h-shadow v-shadow blur spread color inset;

```
box-shadow: 0 0 30px 20px #6699ff inset;        /*内部阴影 */
box-shadow: 0px 0px 45px 10px #9999ff;          /*外部阴影 */
box-shadow: 20px 20px 35px 15px #99ff33;        /* 外部阴影 */
```

第 1 行设置了内部阴影样式为水平、垂直偏移 0px、模糊距离 30px、阴影尺寸 20px、颜色#6699ff。第 2 行设置外部阴影样式为水平、垂直偏移 0px，模糊距离 45px、阴影尺寸 10px、颜色#9999ff。第 3 行设置外部阴影样式为水平、垂直偏移为 20px、模糊距离 35px、阴影尺寸 15px、颜色为#99ff33。

**【例 13-6-2】**CSS3 边框阴影的应用。代码如下，页面效果如图 13-39 所示。

视频讲解

```
<!-- edu_13_6_2.html -->
<!doctype html>
<html lang="en">
    <head>
        <meta charset="UTF-8">
        <title>CSS3边框的应用</title>
        <script type="text/javascript" src="html5shiv.js"></script>
        <link rel="stylesheet" href="css/normalize.css" type="text/css">
        <script type="text/javascript" src="js/prefixfree.min.js"></script>
        <style type="text/css">
                    div{float:left;width:120px;height:120px;margin:50px 80px;background:#dadada;border:6px solid #00cc66;padding:10px;    }
```

```
            #div1{   border-radius:25px;box-shadow: 0 0 30px 20px #6699ff inset;}
            #div2{   border-radius:25px 50px;box-shadow:0px 0px 45px 10px #9999ff;}
            #div3{   border-radius:80px 100px 60px 120px/50px 60px 70px 70px;
                               box-shadow: 20px 20px 35px 15px #99ff33;}
        </style>
    </head>
    <body>
        <h3>CSS3圆角边框、阴影</h3><hr>
        <div id="div1" class=""><p>半径均相同,内部阴影</p></div>
        <div id="div2" class=""><p>左、右对角的半径相同，外部阴影</p></div>
        <div id="div3" class=""><p>每个角水平与垂直半径不同，带水平、垂直偏移
的外部阴影</p></div>
    </body>
</html>
```




图 13-39　CSS3 边框阴影的应用

### 3．border-image 边框图像

通过 CSS3 的 border-image 属性可以创建带有图像的边框，主要参数有三个，分别是图像、剪裁位置、重复性。该属性有五个子属性，其说明如表 13-9 所示。语法如下：

```
border-image:border-image-source
border-image-slice/border-image-width/ border-image-outset border-image- repeat
border-image: url("border.png") 27 27 27 27 fill/27 27 27 27/27px 27px 27px
27px repeat
/* 剪裁和宽度不需要单位，偏移量需要单位。fill表示可选项，指定中间第九块为非透明块。*/
```

**表 13-9　border-image 属性值及说明**

| 值 | 说　明 |
|---|---|
| border-image-source | 规定边框中图像的路径 |
| border-image-slice | 规定图像边框向内偏移，可以是数字或百分比 |
| border-image-width | 规定图像边框的宽度 |
| border-image-outset | 规定边框图像区域超出边框的量 |
| border-image-repeat | 规定图像边框是否应平铺(复制)、铺满(环绕)或拉伸 |

- border-image-source 属性（边框图像）。

默认无边框图像，如果设置边框图像，则使用绝对或相对 url 地址指定边框图像。

```
border-image-source: none |url(image文件);
border-image-source:url("border.png");
```

- border-image-slice 属性（图像切片/剪裁）。

该属性规定图像边框向内偏移，可以是数字或百分比。可以 1～4 个值，类似于 padding 属性的设置方法。语法如下：

```
border-image-slice: number |% |fill;
border-image-slice:27 27 27 27;  /* 边框图像切块9块，每个角为27px*27px*/
```

W3C 指定一个专用位图，图像名称为 border.png，大小为 81px×81px，可以将此图剪裁成小方格为 27px×27px 的九宫格。有四个角、四个边区域和一个中间部分，fill 表示可选项，指定中间第九块为非透明块，不指定说明中间第九块是透明块。如图 13-40 左图所示。border-image-slice 的取值为百分比或数字（默认单位是 px），其中 top/bottom 相对于背景图的高，left/right 相对于背景图的长。如图 13-40 右边图所示。

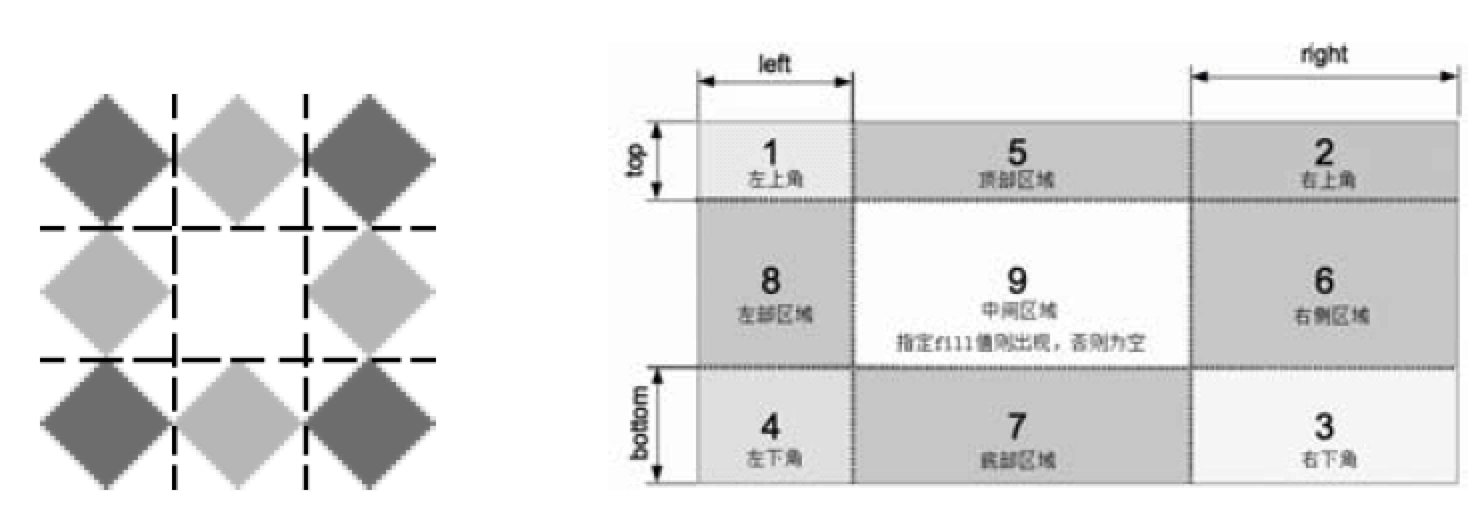



图 13-40　W3C 指定的 81px×81px 位图及九宫格分割法

- border-image-repeat 属性（边框图像重复）。

该属性用于设置边框图像的重复方式。语法如下所示：

```
border-image-repeat: stretch | round | repeat
```

该属性值有三种取值，分别为 stretch（拉伸）、round（环绕）、repeat（复制）。默认值为 stretch。stretch 表示拉伸图像来填充区域；repeat 表示直接用图像来填充区域，填充时图像可能有残缺；round 与 repeat 效果类似，如果无法完整平铺所有图像，则对图像进行缩放以适应区域。

边框将 border-image 分成了九个区域，分别为四个角（border-top-left-image、border-top-right-image、border-bottom-left-image、border-bottom-right-image）、四条边（border-top-image、border-right-image、border-bottom-image、border-left-image）及中间的内容区域，如图 13-41 所示。

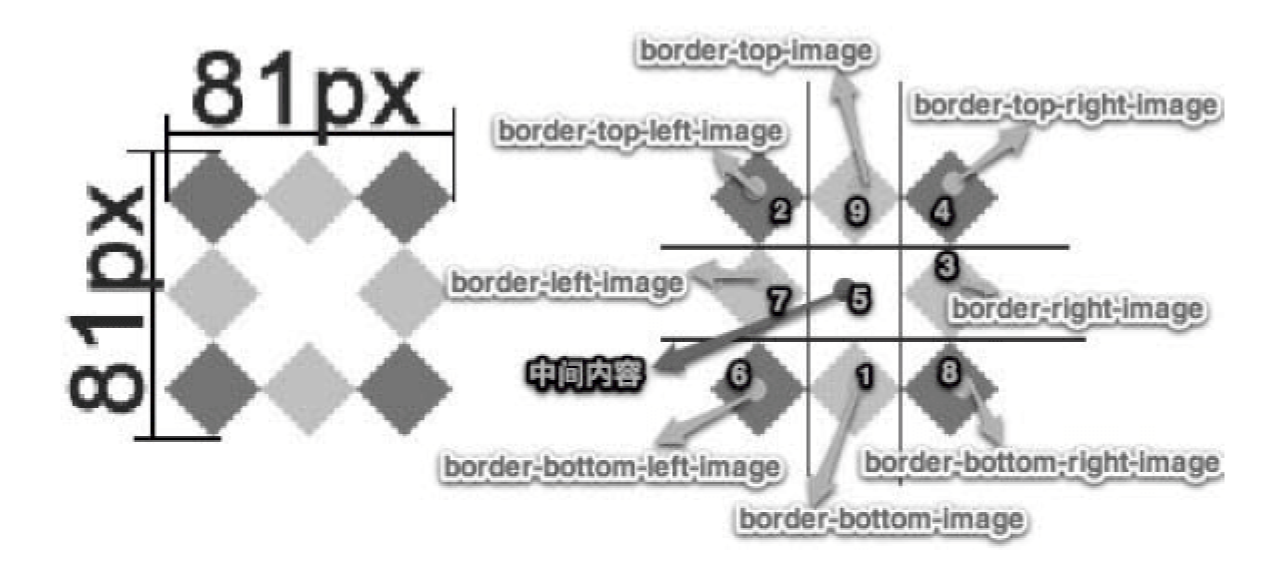



图 13-41　81px×81px 位图及九宫格分割法

边框图像剪裁完后，就可以用裁切的各区域图分别进行绘制。其中，四个角在绘制时会分布在应用元素的四个角上，不会拉伸、平铺或者重复。其他四条边（除了中间 5）的图像分别用来绘制相应的四条边，四条边会应用 border-image-repeat 中设定的排列方式。

- border-image-width 属性（边框图像宽度）。

border-image–width 属性有四个值指定用于把 border 图像区域分为九个部分。它们代表上、右、底部、左、两侧向内距离。如果第四个值被省略，它和第二个是相同的。如果也省略了第三个，那么它和第一个是相同的。如果也省略了第二个，那么它和第一个是相同的。负值是不允许的。

```
border-image-width: number |% ;/* 可以有1～4个值，类似于border-width属性*/
border-image-width:27px 1 10% 27px;/* 边框图像宽度设置为top:27px,right:1倍,
bottom:10%,left:27px */
```

- border-image-outset 属性（图像外凸）。

通过定义距离边框图像区域边缘的向内偏移量，从而指定边框图像的宽度/高度，如图 13-42 所示。注意此属性和 border-width 的区别。

```
border-image-outset: length | number | percentage | auto; /* 可以有1～4个值 */
```

例如，设置 div 的类样式如下，边框图像不向外凸出，页面效果如图 13-42 所示。

```
.box{
    width: 200px; height: 50px;
    border: 54px solid red;    /* 边框宽度54px */
    border-image: url("border.png") 27/27px round;/*边框图像高度与宽度均为27px */}
```

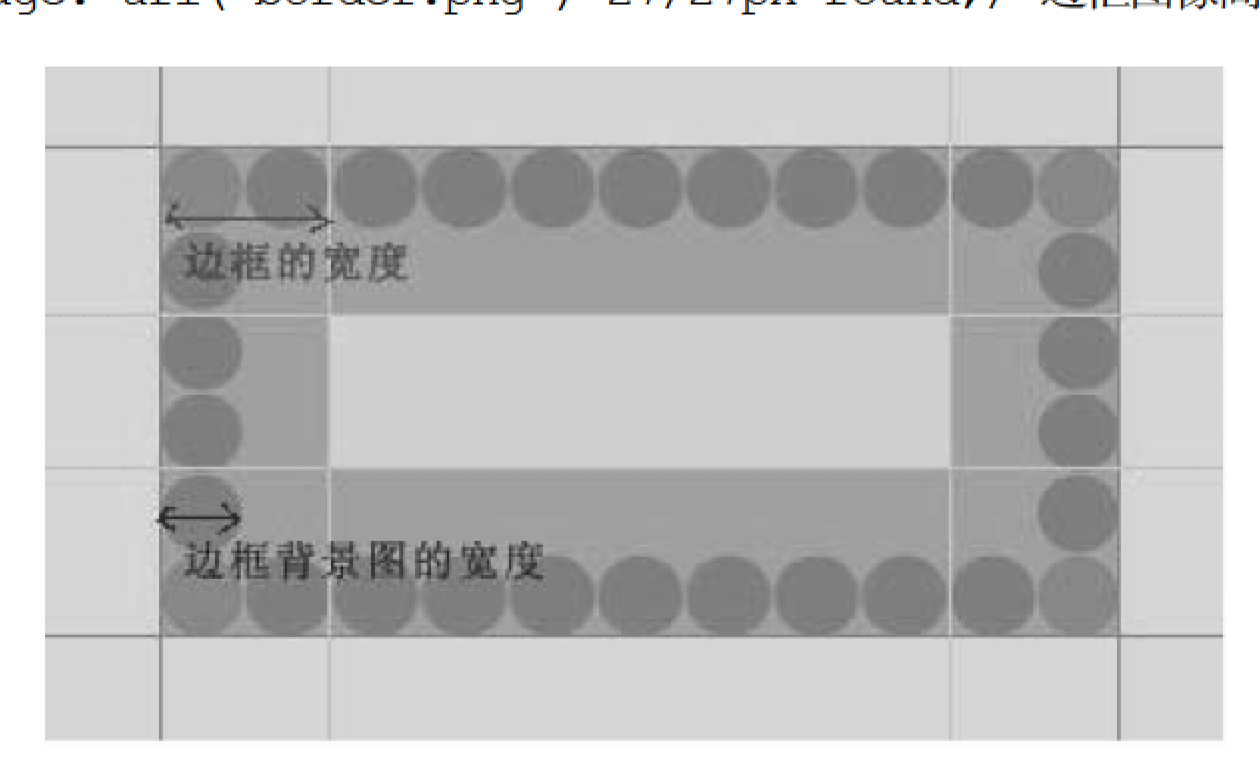



图 13-42　定义边框图像宽度、高度的效果图

例如，设置 div 的类样式如下，边框图像向外凸出，页面效果如图 13-43 所示。

```
.box{
    width: 200px;  height: 50px;
    border: 54px solid red;  /* 边框宽度54px */
    border-image: url("border.png") 27/15px/10px round;  /*指定边框背景图
像宽度为15px、偏移量为10px */}
```

偏移量是建立在边框背景宽度的基础上的，设置偏移量的时候边框背景宽度不能为 0。

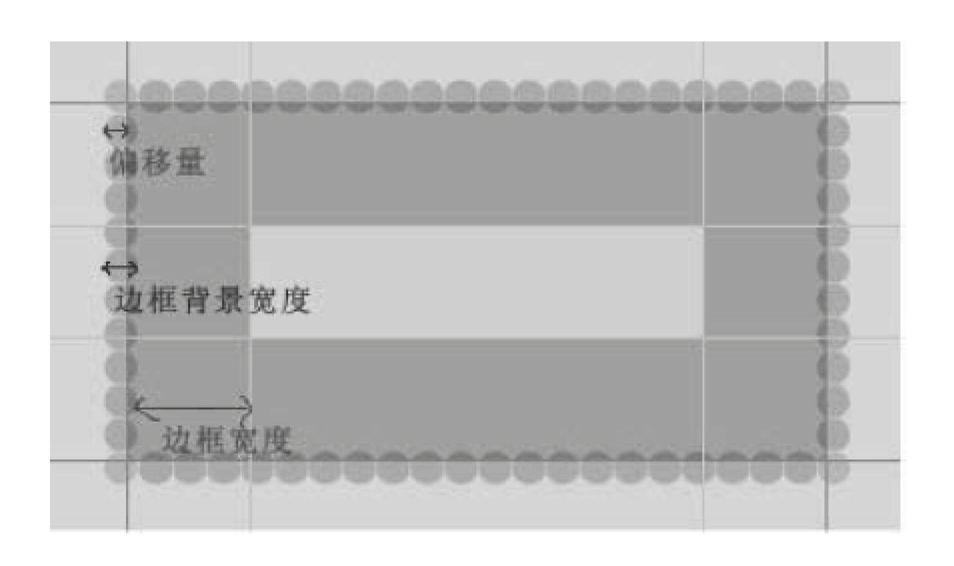



图 13-43　定义边框图像宽度及偏移量的效果图

【例 13-6-3】CSS3 图像边框的应用。代码如下，页面效果如图 13-44 所示。

视频讲解

```
<!-- edu_13_6_3.html -->
<!doctype html>
<html lang="en">
   <head>
       <meta charset="UTF-8">
       <title>CSS3图像边框的应用</title>
       <script type="text/javascript" src="html5shiv.js"></script>
       <link rel="stylesheet" href="css/normalize.css" type="text/css">
       <script type="text/javascript" src="js/prefixfree.min.js"> </script>
       <style type="text/css">
           div{float:left;width:120px;height:120px;margin:30px 30px; background:#dadada;border:1em solid #00cc66;padding:5px;    }
           #div1{border-image:url("border.png") 27 27 stretch;}
           #div2{border-image:url("border.png") 27 27  round;}
           #div3{border-image:url("border.png") 27 27  repeat;}
       </style>
    </head>
    <body>
        <h3>CSS3图像边框的应用</h3><hr>
        <div id="div1" class=""><p>stretch</p></div>
        <div id="div2" class=""><p> round</p></div>
        <div id="div3" class=""><p>repeat</p></div>
        <div id="div4" class="">
        <p>这是原图<img src="border.png" border="0" alt=""></p>
        </div>
    </body>
</html>
```

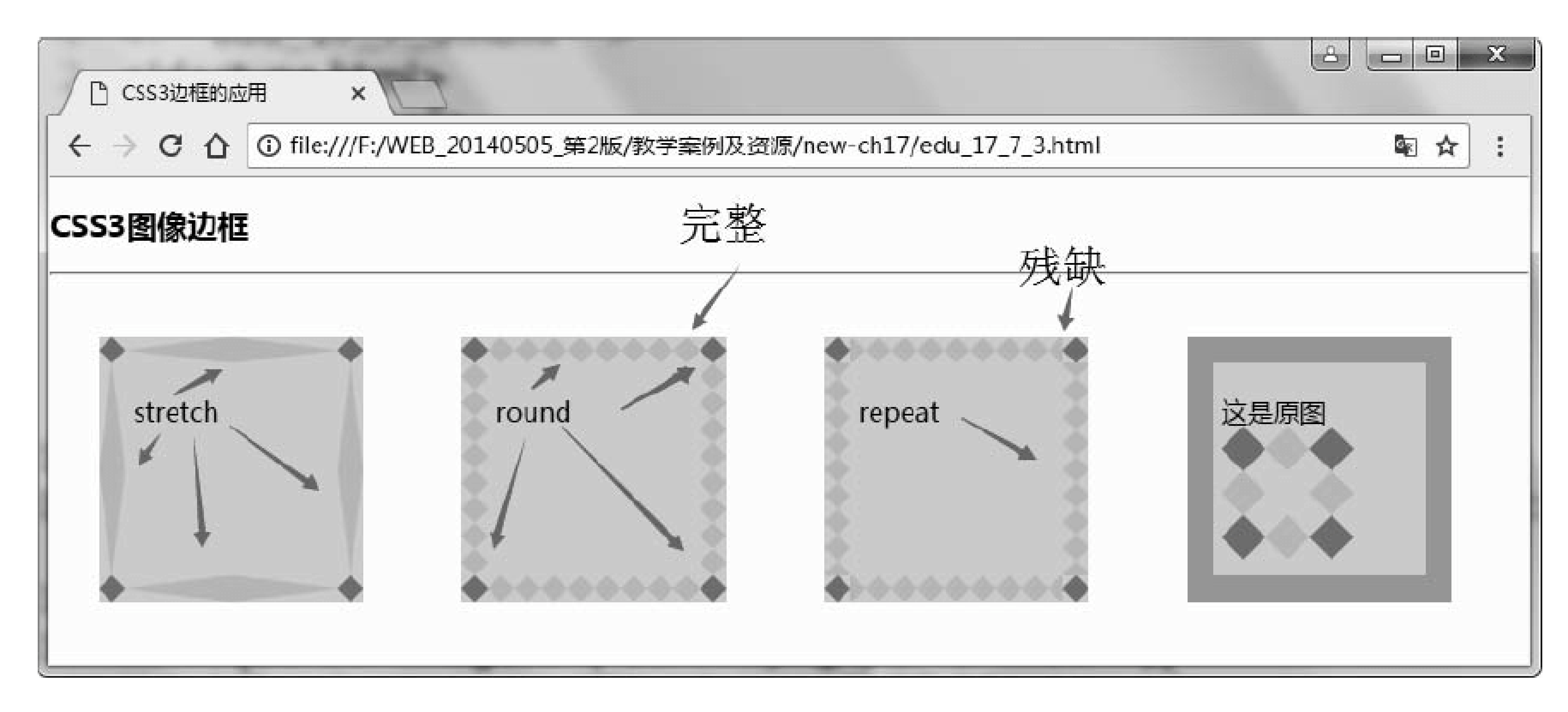



图 13-44　CSS3 图像边框的应用

## 13.6.4　CSS3 转换 transform 属性

通过 CSS3 转换，可以实现对元素进行移动、缩放、转动、拉长或拉伸。所谓转换，就是改变元素形状、尺寸和位置，使其达到另外一种效果的过程。可以对元素进行 2D 或 3D 转换。

**1．CSS3 2D 转换**

CSS3 2D 转换常用方法有 translate()、rotate()、scale()、skew()、matrix()。下面分别介绍每一种方法。

- 位移 translate(x,y)。

translate(x,y)方法的作用是将元素从当前位置根据给定的 x 轴坐标和 y 轴坐标进行移动。x 表示 left，父元素的左边界；y 表示 top，父元素的上边界。translate()方法还提供根据单一轴移动的方法，分别是 translateX()和 translateY()。使用方法如下：

```
transform:translate(50px,50px);     /* 向右移动50px，向下移动50px */
transform:translate(50px,0);        /* 向右移动50px */
transform:translateX(50px);     /* 向右移动50px */
transform:translate(0,50px);    /* 向下移动50px  */
transform:translateY(50px);     /* 向下移动50px  */
```

- 旋转 rotate(deg)。

可以对元素旋转给定的角度，正值为顺时针，负值为逆时针。

```
transform:rotate(deg)           /*  基本语法 */
transform:rotate(10deg)         /*  旋转10°  */
transform:rotate(120deg)        /*  旋转120°  */
```

【例 13-6-4】CSS3 位移与旋转的应用。局部代码如下，页面效果如图 13-45 所示。

视频讲解

```
<!-- edu_13_6_4.html -->
<!doctype html>
<html lang="en">
   <head>
       <meta charset="UTF-8">
       <title>CSS3 2D转换-位移与旋转</title>
       <script type="text/javascript" src="html5shiv.js"></script>
       <link rel="stylesheet" href="css/normalize.css" type="text/css">
       <script type="text/javascript" src="js/prefixfree.min.js"></script>
       <style type="text/css">
           div{width:180px;height:50px;background:#dadada;border:1px solid #00cc66;   }
           #div1{transform:translate(50px,50px);}   /* 位移  */
           #div2{transform:rotate(30deg);}   /* 旋转  */
           #div3{transform:rotate(120deg);}   /* 旋转  */
           td{text-align:left;vertical-align:top;}
       </style>
   </head>
   <body>
       <h3>CSS3 2D转换-位移与旋转</h3><hr>
       <table border="1px" bordercolor="red" width="750px" height="200px">
           <tr>
               <td>
```

```
                    <div id="" class=""><p>这是原div</p></div>
                    <div id="div1" class=""><p>这个div向右移动50px，向下移动50px</p></div>
                </td>
                <td>
                    <div id="" class=""><p>这是原div</p></div>
                    <div id="div2" class=""><p>这个div旋转30度</p></div>
                </td>
                <td>
                    <div id="" class=""><p>这是原div</p></div>
                    <div id="div3" class=""><p>这个div旋转120度</p></div>
                </td>
            </tr>
        </table>
    </body>
</html>
```

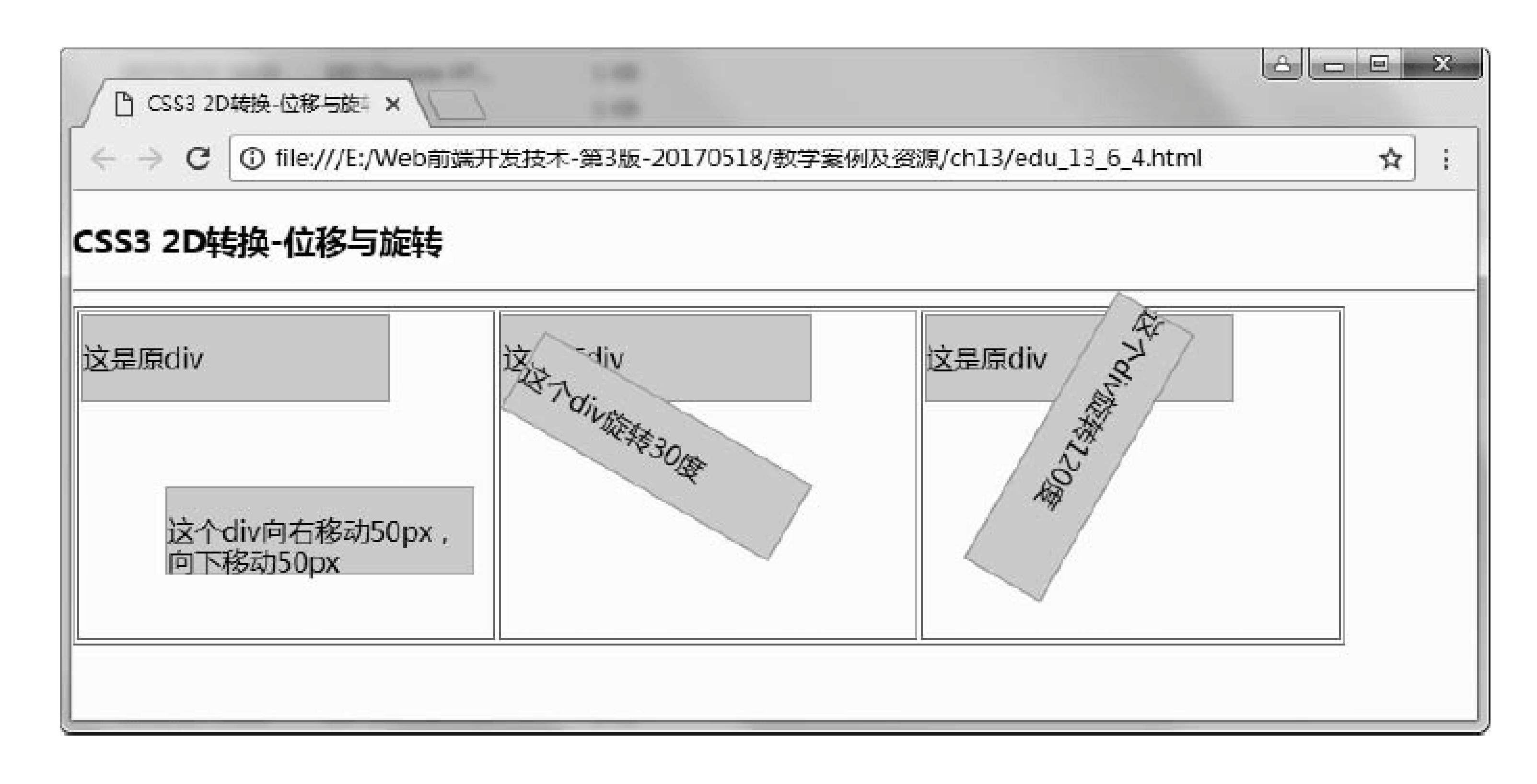



图 13-45　CSS3 2D 位移与旋转的应用

- 缩放 scale(x, y)。

scale(x,y)方法的作用是缩放指定的元素，参数 x 表示元素宽度的缩放倍数，参数 y 表示元素高度的缩放倍数。scale 方法也可以接受负值，当参数 x 为负值时，元素内容会横向倒置；当参数 y 为负值时，元素内容会纵向倒置。

```
transform:scale(x,y);
transform:scale(1,4);
transform:scale(2,2);
```

- 扭曲 skew(deg, deg)。

skew(x,y)方法的作用是将元素翻转（扭曲）给定的角度，参数 x、y 分别表示围绕 x 轴翻转给定的角度、围绕 y 轴翻转给定的角度。

```
transform: skew(deg, deg);
transform: skew(30deg, 30deg);   /*围绕x轴翻转30°，围绕y轴翻转30° */
transform: skew(15deg, 65deg);   /*围绕x轴翻转15°，围绕y轴翻转65° */
```

- 综合转换 matrix(n,n,n,n,n,n)。

matrix()方法和 2D 变换方法合并成一个。matrix()方法是一个综合性的方法，它综合了上述的移动、旋转、缩放等功能。matrix()方法有六个参数，包含旋转、缩放、移动（平移）和倾斜功能。语法如下，参数的作用如下：

```
transform:matrix(scaleX, skewX, skewY, scaleY, translateX, translateY); /* 基本语法*/
transform:matrix(0.866,0.5,-0.5,0.866,20,20);  /* x轴、y轴缩放0.866;x轴、y轴扭曲0.5和-0.5;x轴、y轴位移20px  */
```

参数 1：控制横向缩放，作用于元素的宽度，类似于 scaleX。

参数 2：控制围绕 x 轴翻转的角度，类似于 skewX。

参数 3：控制围绕 y 轴翻转的角度，类似于 skewY。

参数 4：控制纵向缩放，作用于元素的高度，类似于 scaleY。

参数 5：控制元素移动，沿 x 轴进行，类似于 translateX。

参数 6：控制元素移动，沿 y 轴进行，类似于 translateY。

**【例 13-6-5】**CSS3 缩放、扭曲、矩阵综合应用。代码如下，页面效果如图 13-46 所示。

视频讲解

```
<!-- edu_13_6_5.html -->
<!doctype html>
<html lang="en">
   <head>
      <meta charset="UTF-8">
      <title>CSS3 2D转换-扭曲、缩放</title>
      <script type="text/javascript" src="html5shiv.js"></script>
      <link rel="stylesheet" href="css/normalize.css" type="text/css">
      <script type="text/javascript" src="js/prefixfree.min.js"> </script>
      <style type="text/css">
         div{width:100px;height:50px;background:#dadada;border:1px solid #00cc66;    }
         #div1{transform:scale(1.5,1.5);margin:10px auto;}
         #div2{transform:skew(30deg,30deg);margin:10px auto;}
         #div3{transform:matrix(0.866,0.5,-0.5,0.866,20,20);
         /* x轴、y轴缩放0.866;x轴、y轴扭曲0.5和-0.5;x轴、y轴位移20px*/}
         td{text-align:left;vertical-align:top;}
   </style>
   </head>
   <body>
      <h3>CSS3 2D转换-缩放、扭曲、矩阵</h3><hr>
      <table border="1px" bordercolor="red" width="750px" height="200px">
         <tr>
            <td>
               <div id="" class=""><p>这是原div</p></div>
               <div id="div1" class=""><p>这个div缩放1.5倍</p></div>
            </td>
            <td>
               <div id="" class=""><p>这是原div</p></div>
               <div id="div2" class=""><p>这个div扭曲方法</p></div>
            </td>
            <td>
               <div id="" class=""><p>这是原div</p></div>
               <div id="div3" class=""><p>这是div采用matrix方法</p></div>
            </td>
         </tr>
      </table>
   </body>
</html>
```




图 13-46　CSS3 2D 缩放、扭曲、矩阵综合应用

### 2. CSS3 3D 转换

CSS3 可以使用 3D 转换来对元素进行格式化。常用的 3D 转换方法有 rotateX()、rotateY()。

- 旋转 rotateX()方法。

通过 rotateX()方法，元素围绕其 X 轴以给定的度数进行旋转。

- 旋转 rotateY()方法。

通过 rotateY()方法，元素围绕其 Y 轴以给定的度数进行旋转。

```
transform: rotateX(angle);   /*  X轴方向旋转一定角度 */
transform: rotateY(angle);   /* Y轴方向旋转一定角度  */
#div1{transform:rotateX(120deg);}
#div2{transform:rotateY(120deg);margin:10px auto;}
```

**【例 13-6-6】**CSS3 3D 旋转的应用。代码如下，页面效果如图 13-47 所示。

视频讲解

```
<!-- edu_13_6_6.html -->
<!doctype html>
<html lang="en">
   <head>
      <meta charset="UTF-8">
      <title>CSS3 3D转换</title>
      <script type="text/javascript" src="html5shiv.js"></script>
      <link rel="stylesheet" href="css/normalize.css" type="text/css">
      <script type="text/javascript" src="js/prefixfree.min.js"></script>
      <style type="text/css">
         div{width:150px;height:80px;background:#dadada;border:1px solid #00cc66;    }
         #div1{transform:rotateX(120deg);}
         #div2{transform:rotateY(120deg);margin:10px auto;}
         td{text-align:left;vertical-align:top;}
      </style>
   </head>
   <body>
      <table border="1px" align="center" width="450px" height="200px">
         <caption><h3>CSS3 3D转换</h3></caption>
         <tr>
            <td>
               <div id="" class=""><p>这是原div</p></div>
               <div id="div1" class=""><p>沿X轴旋转这个div</p></div>
            </td>
            <td>
               <div id="" class=""><p>这是原div</p></div>
```

```
                    <div id="div2" class=""><p>沿Y轴旋转这个div</p></div>
                </td>
            </tr>
        </table>
    </body>
</html>
```




图 13-47　CSS3 3D 旋转应用

## 13.6.5　CSS3 过渡 transition 属性

通过 CSS3 过渡，可以在不使用 Flash 动画或 JavaScript 的情况下，实现元素从一种样式到另一种样式的转变效果。

CSS3 过渡是元素从一种样式逐渐改变为另一种的效果。要实现这种效果，需要设置两个因素，分别是指定要添加效果的 CSS 属性、指定效果的持续时间。如果未指定的期限，transition 将没有任何效果，因为默认值是 0。

transition 属性是一个复合属性，它有四个过渡属性，如表 13-10 所示。语法如下：

```
transition: property duration timing-function delay;
transition: width 2s;  /* 宽度上过渡2s */
```

**表 13-10　transition 属性的值及说明**

| 值 | 说　　明 |
|---|---|
| transition-property | 规定设置过渡效果的 CSS 属性的名称 |
| transition-duration | 规定完成过渡效果需要多少秒或毫秒 |
| transition-timing-function | 规定速度效果的速度曲线 |
| transition-delay | 定义过渡效果何时开始 |

- transition-property。

transition-property 属性指定 CSS 属性的 nametransition 效果（transition 效果时将会启动指定的 CSS 属性的变化）。一个转场效果，通常会出现在当用户将鼠标悬停在一个元素上时。

```
transition-property: none|all| property;
transition-property: width;   /* width属性上转场 */
```

- transition-duration。

transition-duration 属性规定完成过渡效果需要花费的时间（以秒或毫秒计）。

```
transition-duration: time;
transition-duration: 3s;
```

- transition-timing-function。

```
transition-timing-function: linear|ease|ease-in|ease-out|ease-in-out| cubic-bezier(n,n,n,n);
```

其 6 种取值分别为匀速 linear、逐渐变慢 ease、加速 ease-in、减速 ease-out、加速后减速 ease-in-out、贝塞尔曲线 cubic-bezier(n,n,n,n)，如表 13-11 所示。

**表 13-11　transition-timing-function 的值及说明**

| 值 | 说　明 |
|---|---|
| linear | 规定以相同速度从开始至结束的过渡效果(cubic-bezier(0,0,1,1)) |
| ease | 规定以慢速开始、变快、慢速结束的过渡效果。<br>类似于 cubic-bezier(0.25,0.1,0.25,1) |
| ease-in | 规定以慢速开始的过渡效果( cubic-bezier(0.42,0,1,1)) |
| ease-out | 规定以慢速结束的过渡效果(cubic-bezier(0,0,0.58,1)) |
| ease-in-out | 规定以慢速开始和结束的过渡效果(cubic-bezier(0.42,0,0.58,1)) |
| cubic-bezier(n,n,n,n) | 在 cubic-bezier 函数中定义自己的值。可能的值在 0～1 之间 |

- transition-delay。

transition-delay 属性指定何时将开始切换效果。指以秒(s)或毫秒(ms)为单位。

```
transition-delay: time;
transition-delay: 2s;
```

**【例 13-6-7】**CSS3 过渡与转换综合的应用。代码如下，页面效果如图 13-48 和图 13-49 所示。

视频讲解

```
<!-- edu_13_6_7.html -->
<!doctype html>
<html lang="en">
    <head>
        <meta charset="UTF-8">
        <title>CSS3 过渡</title>
        <script type="text/javascript" src="html5shiv.js"></script>
        <link rel="stylesheet" href="css/normalize.css" type="text/css">
        <script type="text/javascript" src="js/prefixfree.min.js"></script>
         <style>
             div{width:100px;height:50px;background:#009999;color:white;font-weight:bold;
             transition:width 2s,height 2s,transform 2s;/* 3个属性过渡 */}
             #div1 {transition-timing-function: linear;}
             #div2 {transition-timing-function: ease;}
             #div3 {transition-timing-function: ease-in;}
             #div4 {transition-timing-function: ease-out;}
             #div5 {transition-timing-function: ease-in-out;}
             div:hover{width:200px; height:100px;transform:rotate(60deg);
             /* 盘旋时过渡+旋转 */}
         </style>
     </head>
     <body>
         <h3>CSS3 过渡transition与transform综合应用</h3><hr color="red">
         <div id="div1" style="top:100px">linear</div>
         <div id="div2" style="top:150px">ease</div>
         <div id="div3" style="top:200px">ease-in</div>
```

```
        <div id="div4" style="top:250px">ease-out</div>
        <div id="div5" style="top:300px">ease-in-out</div>
        <p>请把鼠标指针移动到红色的div元素上，就可以看到<mark>过渡和转换</mark>
的效果。</p>
    </body>
</html>
```




图 13-48　CSS3 过渡初始状态图




图 13-49　CSS3 过渡盘旋时状态图

## 13.6.6　CSS3 动画 animation

所谓 CSS3 动画，就是指元素从一种样式逐渐变化为另一种样式的效果。通过 CSS3 的@keyframes（关键帧）规则，可以创建动画，从而取代动画图片、Flash 动画以及 JavaScript 编写的动画。在@keyframes 中规定某项 CSS 样式，就能创建由当前样式逐渐改为新样式的动画效果。

### 1．CSS3 动画 animation 基本语法

animation 是一个复合属性，语法如下，其属性及说明如表 13-12 所示。

```
animation:animation-name|animation-duration|animation-timing-function|a
nimation-delay | animation-iteration-count| animation-direction
```

**表 13-12　CSS3 动画属性及说明**

| 属　性 | 说　明 |
|---|---|
| @keyframes | 规定动画 |
| animation | 所有动画属性的复合属性，除了 animation-play-state 属性 |
| animation-name | 规定@keyframes 动画的名称 |
| animation-duration | 规定动画完成一个周期所花费的秒或毫秒，默认是 0 |
| animation-timing-function | 规定动画的速度曲线，默认是 ease，其他与 transition-timing-function 属性值相同 |
| animation-delay | 规定动画何时开始，默认是 0 |
| animation-iteration-count | 规定动画被播放的次数 n（值为 1（默认）、infinite） |
| animation-direction | 规定动画是否在下一周期逆向地播放（值为 normal（默认）、alternate） |
| animation-play-state | 规定动画是否正在运行或暂停，其值为 running（默认）、paused |
| animation-fill-mode | 规定对象动画时间之外的状态（其值为 none、forwards、backwards、both） |

### 2．@keyframes 规则定义

采用@keyframes 规则创建动画，需要将它绑定到一个 CSS 的选择器，否则动画不会

有任何效果。定义至少以下两项 CSS3 动画属性，即可将动画绑定到选择器：规定动画的名称、规定动画的时长。

@keyframes 基本语法：

```
@keyframes myAnimation {
    from {Properties:Properties value;  }
    Percentage {Properties:Properties value; }
    to {Properties:Properties value;  }
  }
```

或者全部写成百分比的形式：

```
@keyframes myAnimation {
  0% {Properties:Properties value; }
  Percentage {Properties:Properties value; }
  100% {Properties:Properties value; }
}
```

语法说明：其中 myAnimation 是一个动画名称，最好有特定的含义。用百分比来规定变化发生的时间，或用关键词 from 和 to，等同于 0%和 100%。0% 是动画的开始，100% 是动画的完成。Percentage 是百分比值，我们可以添加许多个这样的百分比，Properties 为 CSS 的属性名，例如 width、heigth、background 等；value 就是对应的属性的值。选择器 from 和 to 分别对应选择器 0%和 100%。Internet Explorer 9 及更早 IE 版本是无效的。

为了得到最佳的浏览器支持，至少应该定义 0%和 100%选择器两个选择器。中间状态也可以根据需要增加 n 个选择器，并定义样式，来完成动画。

**3. @keyframes 规则的绑定**

绑定动画名称（例如 myAnimation）到某个元素(div)的样式上，并指定时长。格式如下：

```
div{
    animation: myAnimation 8s;
    -moz-animation: myAnimation 8s;             /* Firefox */
    -webkit-animation: myAnimation 8s;          /* Safari 和 Chrome */
    -o-animation: myAnimation 8s;               /* Opera */
}
```

以 Safari 和 Chrome 浏览器为例，说明动画的设置方法，代码如下所示。

```
div{
    /* 设置图层基本样式 */
    width:100px;height:100px;background:red;position:relative;
    /* 设置标准动画子属性  */
    animation-name:myMOve;
    animation-duration:5s;
    animation-timing-function:linear;
    animation-delay:2s;
    animation-iteration-count:infinite;
    animation-direction:alternate;
    animation-play-state:running;
    /* 仅以Safari and Chrome浏览器为例，其余类似。 */
    -webkit-animation-name: myMOve;
    -webkit-animation-duration:5s;
```

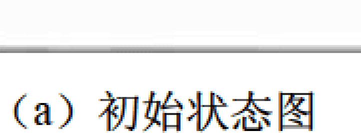
```
        -webkit-animation-timing-function:linear;
        -webkit-animation-delay:2s;
        -webkit-animation-iteration-count:infinite;
        -webkit-animation-direction:alternate;
        -webkit-animation-play-state:running;
    }
    @keyframes myMOve
    {   /* 定义不同关键帧的样式 */
       0%  {background:red; left:0px; top:0px;}
       25% {background:yellow; left:200px; top:0px;}
       50% {background:blue; left:200px; top:200px;}
       75% {background:green; left:0px; top:200px;}
       100% {background:red; left:0px; top:0px;}
    }
    @-webkit-keyframes myfirst /* 仅以Safari and Chrome为例 */
    {   /* 定义不同关键帧的样式 */
       0%  {background:red; left:0px; top:0px;}
       25% {background:yellow; left:200px; top:0px;}
       50% {background:blue; left:200px; top:200px;}
       75% {background:green; left:0px; top:200px;}
       100% {background:red; left:0px; top:0px;}
    }
```

**【例 13-6-8】**CSS3 动画的应用。代码如下，页面效果如图 13-50 所示。100px×100px 的 div 沿 300px×300px 的矩形对角线运动。边运动边改变背景颜色，从红色到蓝色过渡。设置三个场景切换：初始状态为红色背景，中间状态为绿色背景，最后状态为蓝色背景。同时考虑到不同浏览器的兼容性，代码中增加了针对不同浏览器编写的样式效果。

视频讲解

```
<!-- edu_13_6_8.html -->
<!doctype html>
<html lang="en">
   <head>
      <meta charset="UTF-8">
      <title>CSS3动画</title>
      <style>
         div{width:100px;height:100px;background:red;position:relative;
            animation:mymove 5s ;    color:white;
            -moz-animation:mymove 5s infinite;    /* Firefox */
            -webkit-animation:mymove 5s infinite;    /* Safari and Chrome */
            -o-animation:mymove 5s infinite;     /* Opera */
         }
         @keyframes mymove
         {
            from,0% {left:0px;background:red;top:0px;}
            50%   {left:100px;background:green;top:100px;}
            to,100%{left:200px;background:blue;top:200px;}
         }
         @-webkit-keyframes mymove    /* Safari 与 Chrome */
         {
            from,0% {left:0px;background:red;top:0px;}
            50%   {left:100px;background:green;top:100px;}
            to,100%{left:200px;background:blue;top:200px;}
         }
         @-moz-keyframes mymove   /* Firefox */
         {
            from,0% {left:0px;background:red;top:0px;}
            50%   {left:100px;background:green;top:100px;}
```

```
                to,100%{left:200px;background:blue;top:200px;}
            }
            @-o-keyframes mymove   /* Opera */
            {
                from,0% {left:0px;background:red;top:0px;}
                50%   {left:100px;background:green;top:100px;}
                to,100%{left:200px;background:blue;top:200px;}
            }
        </style>
    </head>
    <body>
        <h3>CSS3动画-沿矩形对角线运动</h3><hr>
        <div>我在运动！</div>
    </body>
</html>
```

（a）初始状态图



（b）中间状态图




（c）最后初始状态图

图 13-50　CSS3 动画状态图

## 13.6.7　CSS3 多列属性

使用 CSS3 多列属性可以创建多个列来对文本进行布局，如同编辑报纸和杂志一样。但 IE9 以及更早的版本不支持多列属性。常用的 CSS3 多列属性主要有 column-count、column-gap、column-rule 等，如表 13-13 所示。

**表 13-13　CSS3 多列属性值及说明**

| 属　　性 | 说　　明 |
|---|---|
| columns | 规定设置 column-width 和 column-count 的复合属性 |
| column-count | 规定元素应该被分隔的列数 |
| column-width | 规定列的宽度 |
| column-fill | 规定如何填充列 |
| column-gap | 规定列之间的间隔 |
| column-rule | 设置所有 column-rule-*属性的复合属性 |
| column-rule-width | 规定列之间规则的宽度 |
| column-rule-style | 规定列之间规则的样式 |
| column-rule-color | 规定列之间规则的颜色 |
| column-span | 规定元素应该横跨的列数 |

**基本语法**

```
columns: column-width column-count; /* 复合属性*/
```

```
    column-count: number|auto
    column-width: auto|length;
    column-rule: column-rule-width column-rule-style column-rule-color;/* 复合属性*/
    column-rule-width: thin|medium|thick|length;
    column-rule-style: none|hidden|dotted|dashed|solid|double|groove| ridge|
inset|outset;
    column-rule-color: color;
    column-gap: length|normal;
    column-fill: balance|auto; /* balance列长短平衡; auto列顺序填充*/
```

**【例 13-6-9】**CSS3 多列属性的应用。代码如下，页面效果如图 13-51 所示。

视频讲解

```
<!-- edu_13_6_9.html -->
<!doctype html>
<html lang="en">
    <head>
        <meta charset="UTF-8">
        <title>CSS3多列属性的应用</title>
        <style>
            p{
                text-indent:2em;
                column-count:3; /* 设置列数 */
                column-gap:50px; /* 设置列间隙 */
                column-rule:4px outset #ff0000; /* 设置列宽度、线型、颜色 */
            }
            h2{
                column-span:all; /* 设置标题跨所有列 */
                text-align:center;background:#99ff99;
                height:40px;font-size:28px;padding:6px auto;}
        </style>
    </head>
    <body>
        <h2>HTML 5 简介</h2>
        <p>HTML标准自1999年12月发布的HTML4.01后，后继的HTML5和其他标准被束之高阁，为了推动Web标准化运动的发展，一些公司联合起来，成立了一个叫作 Web Hypertext Application Technology Working Group （Web超文本应用技术工作组 -WHATWG） 的组织。WHATWG 致力于 Web 表单和应用程序，而W3C（World Wide Web Consortium，万维网联盟） 专注于XHTML2.0。在 2006 年，双方决定进行合作，来创建一个新版本的 HTML。<br>
        HTML5草案的前身名为 Web Applications 1.0，于2004年被WHATWG提出，于2007年被W3C接纳，并成立了新的 HTML 工作团队。<br>
        HTML 5 的第一份正式草案已于2008年1月22日公布。HTML5 仍处于完善之中。然而，大部分现代浏览器已经具备了某些 HTML5 支持。<br>
        2012年12月17日，万维网联盟（W3C）正式宣布凝结了大量网络工作者心血的HTML5规范已经正式定稿。根据W3C的发言稿称："HTML5是开放的Web网络平台的奠基石。"<br>
        2013年5月6日， HTML 5.1 正式草案公布。该规范定义了第五次重大版本，第一次要修订万维网的核心语言：超文本标记语言（HTML）。在这个版本中，新功能不断推出，以帮助Web应用程序的作者，努力提高新元素互操作性。<br>
        本次草案的发布，从2012年12月27日至今，进行了多达近百项的修改，包括HTML和XHTML的标签，相关的API、Canvas等，同时HTML5的图像img标签及svg也进行了改进，性能得到进一步提升。<br>
        支持Html5的浏览器包括Firefox（火狐浏览器），IE9及其更高版本，Chrome（谷歌浏览器），Safari，Opera等；国内的傲游浏览器（Maxthon），以及基于IE或Chromium（Chrome的工程版或称实验版）所推出的360浏览器、搜狗浏览器、QQ浏览器、猎豹浏览器等国产浏览器同样具
```

```
备支持HTML5的能力。</p>
    29      </body>
    30 </html>
```




图 13-51　CSS3 多列属性的应用

## 13.6.8　CSS3 文本效果

CSS3 新定义了多个文本特性，常用的文本属性有文本阴影 text-shadow、控制换行 word-wrap、文本溢出 text-overflow、文本换行 text-wrap 等。

**1．文本阴影 text-shadow 属性**

1）基本语法

```
text-shadow: h-shadow v-shadow blur color;
text-shadow:2px 2px 8px #FF0000;
```

2）语法说明

text-shadow 属性向文本添加一个或多个阴影。该属性是空格分隔的阴影列表，其中 h-shadow 定义水平阴影，允许负值，必需；v-shadow 定义垂直阴影，允许负值，必需；blur 可选。模糊的距离。color 可选。阴影的颜色。省略的长度是 0。

**2．文本换行 text-wrap 属性**

1）基本语法

```
text-wrap: normal|none|unrestricted|suppress;
```

2）语法说明

text-wrap 属性指定文本换行规则。所有浏览器目前均不支持此属性。

**3．控制换行 word-wrap 属性**

1）基本语法

```
word-wrap: normal|break-word;
```

2）语法说明

word-wrap 自动换行属性允许强制文本进行换行，即使这意味着会对单词进行拆分。该属性有两个值，分别为 normal、break-word。其中 normal 表示只在允许的断字点换行（浏览器保持默认处理）；break-word 表示在长单词或 URL 地址内部进行换行。

#### 4．文本溢出 text-overflow 属性

1）基本语法

```
text-overflow: clip|ellipsis|string;
```

2）语法说明

text-overflow: 属性规定当文本溢出包含元素时发生的事情。该属性有三个属性值，分别为 clip、ellipsis、string。其中 clip 表示修剪文本；ellipsis 表示显示省略符号来代表被修剪的文本；string 表示使用给定的字符串来代表被修剪的文本。

**【例 13-6-10】**CSS3 文本效果属性的应用。代码如下，页面效果如图 13-52 所示。

视频讲解

```
<!-- edu_13_6_10.html -->
<!doctype html>
<html lang="en">
   <head>
      <meta charset="UTF-8">
      <title>Document</title>
      <style>
        h2{text-align:center;background:#99ccff;padding:5px auto;}
        h1 {text-shadow:2px 2px 8px #FF0000;/* 设置文本阴影 */}
        p.test{width:15em; border:1px solid #000000;
               word-wrap:break-word;/* 设置自动换行*/}
        div.test{white-space:nowrap;/* 规定文本不进行换行 */
                width:12em;overflow:hidden; border:1px solid #000000;}
      </style>
   </head>
   <body>
   <h2>CSS3文本效果</h2>
      <h1>具有模糊效果的文本阴影</h1>
      <p>【未设置换行和宽度的段落】This paragraph contains a very long word: thisisaveryveryveryveryveryverylongword. The long word will break and wrap to the next line.</p>
      <p class="test">【设置强制换行和宽度的段落】This paragraph contains a very long word: thisisaveryveryveryveryveryverylongword. The long word will break and wrap to the next line.</p>
      <h3>下列div 包含长文本，都能正常显示</h3>
      <div id="" class="">
         HTML 5 的第一份正式草案已于2008年1月22日公布。HTML5 仍处于完善之中。然而，大部分现代浏览器已经具备了某些 HTML5 支持。
      </div>
      <h3>下面两个div包含无法在框中容纳的长文本。不能完全显示，文本被修剪了。</h3><hr>
      <p>下列div使用 "text-overflow:ellipsis" : </p>
       <div class="test" style="text-overflow:ellipsis;">HTML 5 的第一份正式草案已于2008年1月22日公布。HTML5 仍处于完善之中。然而，大部分现代浏览器已经具备了某些 HTML5 支持。</div>
      <h3>下列div使用 "text-overflow:clip": </h3>
      <div class="test" style="text-overflow:clip;">HTML 5 的第一份正式草案已于2008年1月22日公布。HTML5 仍处于完善之中。然而，大部分现代浏览器已经具备了某些 HTML5 支持。</div>
   </body>
</html>
```




图 13-52　CSS3 文本效果的应用



视频讲解

## 13.7　综 合 实 例

以“HUAWEI CONNECT 2016 全联接大会”的会议注册页面为例，采用 HTML5 构建页面。代码如下，页面效果如图 13-53 所示。设计要求如下：




图 13-53　HUAWEI CONNECT 2016 全联接大会注册页面

（1）整个页面采用 article 标记构架。使用 header、footer、hgroup、figure、figcaption、form、fieldset 等标记来进行页面布局。

（2）会议注册页面。采用 fieldset、legend 进行表单元素分组。在其中分别采用 input 类型为 text、email、tel、date、submit、reset 等来布局页面，注册姓名、报告题目、工作单位等文本输入域不能为空。

（3）footer 部分中宾馆信息采用 address 标记进行布局。

```
<!-- edu_13_7_1.html -->
<!doctype html>
<html lang="en">
   <head>
      <meta charset="UTF-8">
      <title>HTML5表单</title>
      <style type="text/css">
         img:hover{transform:rotate(30deg);/*鼠标盘旋时，旋转30° */}
      </style>
   </head>
   <body>
      <article style="margin:20px auto;width:850px;height:900px; background:#eeeeee;padding:30px;">
         <header>
            <hgroup>
               <h1>塑造云时代，联接企业的现在与未来</h1>
               <h3>HUAWEI CONNECT 2016全联接大会</h3>
               <p style="text-indent:2em;">[中国，上海，<time datetime="2016-09-01">2016年09月1日</time>]在HUAWEI CONNECT 2016全联接大会上，华为IT产品线总裁郑叶来发表了“塑造云时代，联接企业的现在与未来”主题演讲，重磅发布了基于FusionCloud的31款云服务、FusionStorage 6.0云存储与FusionStage PaaS平台，并与现场的客户、合作伙伴、媒体、分析师等分享了华为在塑造云时代的过程中如何帮助企业加速云化，挖掘更多商业价值。</p>
               <figure style="text-align:center;">
                  <img src="huawei_it.jpg" width="500" border="0" ><br>
                     <figcaption>华为IT产品线总裁郑叶来发表主题演讲</figcaption>
               </figure>
            </hgroup>
         </header>
         <form method="post" action="">
            <fieldset style="text-align:center;margin:10px 40px;">
               <legend>会议注册页面</legend>
               注册姓名：<input type="text" name="" required> 报告题目：
               <input type="text" name="" required><br> 工作单位：
               <input type="text" name="" required> 手机号码：
               <input type="tel"><br> 电子邮件：
               <input type="email" name=""> 报到时间：
               <input type="date" /><br>
               <input type="submit" value="注册"><input type="reset">
            </fieldset>
         </form>
         <footer>
            <p>住宿信息：请3月14日以后注册的代表自行预订会议期间的住宿。会务邮箱phylab@fudan.edu.cn。</p>
            <p>给您推荐的宾馆:<address>邯郸路100号如家快捷酒店(复旦店)酒店电话021-51055577国定路428号 ;<br> 同舟大厦 酒店电话 021-65110356 。两家宾馆距离学校
```

```
都非常近，步行5分钟之内。</address></p>
39            </footer>
40         </article>
41     </body>
42 </html>
```

代码中第 8 行给图像定义了鼠标盘旋时样式（旋转 30°）。第 12～40 行为一个 article 标记区。第 13～23 行为 header 标记区，采用 hgroup 标记来布局，其中第 18～21 行采用 figure 和 figcaption 标记组合标注图像及标题。第 24～35 行为表单注册区域，分别对相关输入域进行合法性检查设置。第 36～39 行为 footer 标记区域。

# 本 章 小 结

本章介绍了 HTML5 新特性和一些基础的 HTML5 的应用。重点讲述 HTML5 的新增属性、新增表单属性、新增表单的 input 类型、媒体元素（视频、音频）等方面的知识和程序设计技巧。

HTML5 新增了 header、nav、article、section、aside、footer 等结构元素,使用这些语义的标记构建网页更为方便、快捷。HTML5 新增的其他页面元素也极大地丰富了页面内容与表现，结合 JavaScript 脚本能够设计具有更好的用户体验的网站。HTML5 技术在移动互联网时代会具有更加杰出的表现。

运用 CSS3 新增转换、过渡和动画特性可以增强页面的表现效果。运用 CSS3 多列属性、文本效果属性可以美化页面排版效果。

# 练习与实验

## 练习 13

**1．选择题**

（1）HTML5 之前的 HTML 版本是（　　）。

A．HTML4.9　　B．HTML4　　C．HTML4.01　　D．HTML4.1

（2）HTML5 的正确 doctype 是（　　）。

A．<!DOCTYPE html>

B．<!DOCTYPE HTML5>

C．<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 5.0//EN" "http://www.w3.org/TR/html5/strict.dtd">

D．以上都不是

（3）在 HTML5 中，属于组合标题标记是（　　）。

A．<group>　　B．<header>　　C．<headings>　　D．<hgroup>

（4）用于播放 HTML5 视频文件的正确 HTML5 元素是（　　）。

A．<media>　　B．<audio>　　C．<video>　　D．<movie>

（5）在 HTML5 中，规定输入字段是必填的属性是（　　）。

A．required　B．formvalidate　C．validate　D．placeholder

（6）下列属于输入类型为定义滑块控件的是（　　）。

A．search　B．controls　C．slider　D．range

（7）下列输入类型用于定义周和年控件（无时区）的是（　　）。

A．date　B．week　C．year　D．time

（8）下列 HTML5 元素用于显示已知范围内的标量测量的是（　　）。

A．<gauge>　B．<range>　C．<measure>　D．<meter>

（9）下列属性中表示 CSS3 的过渡的属性是（　　）。

A．animation　B．transform　C．transition　D．box-shadow

（10）下列属性中能够设置圆角边框的属性的是（　　）。

A．box-shadow　B．border-image　C．border-style　D．border-radius

（11）下列属性中不是过渡 transition 子属性的是（　　）。

A．transition-property　B．transition-delay

C．transition-play　D．transition-duration

（12）下列选项中定义动画 animation 的关键帧的是（　　）。

A．@keyframes　B．keyfrmaes　C．@import url()　D．@keyframe

**2．填空题**

（1）数据列表选项 datalist 标记通常与______标记结合在一起使用，通过该标记______属性与 datalist 标记的_________属性关联。

（2）HTML5 新增媒体元素除了通过______属性可以加载媒体文件 URL 外，还可以通过__________标记加载不同格式的媒体文件，以满足浏览器支持的需要。

（3）HTML5 新增________类型的 input 元素可以拾取颜色；新增_________类型的 input 元素可以对邮箱进行自动验证；新增_________类型的 input 元素可以产生滑动条控件；新增_________类型的 input 元素可以产生带有微调按钮的输入域。

（4）HTML5 新增_________表单元素可以产生数据加密；新增_________表单元素可以产生不同类型的输出。新增__________表单元素可以定义选项列表。

**3．简答题**

（1）简述 HTML5 文档结构的基本组成。

（2）简述 HTML5 的八大特性。

（3）简述 CSS3 动画与 CSS3 过渡的区别。

## 实验 13

1．采用 HTML5 和 CSS3 多列设计一个简易的 HTML5 页面，效果如图 13-54 所示。具体要求如下：

（1）整个页面采用 article 标记构架。

（2）使用 header、figure、figcaption、footer、hgroup 等标记来进行页面布局。

（3）标题采用 CSS 文本阴影。

（4）程序名称为 exp_13_1.html。

提前探班：HUAWEI CONNECT 2016大透析

为什么华为要自主办HUAWEI CONNECT旗舰大会？




▲华为常务董事、战略Marketing总裁徐文伟

徐文伟在采访中把HC2016大会的举办归结于华为历史的发展，他表示：“华为发展的过程，我认为可以分为三个阶段。第一个阶段，华为主要参加行业组织和外部机构举办的活动。比如每年的巴展、CeBIT和interop等等。第二个阶段，随着公司的发展，华为除了在继续参与这些活动之外，也开始自行组织一些有特定主题、面向特定受众的活动。比如HCC(华为云计算大会)、HNC(华为网络大会)、HDC(华为开发者大会)等。第三个阶段，华为开始筹划组织HUAWEI CONNECT这样的旗舰大会。华为希望通过构筑平台开放能力，最终释放整个ICT产业的生产力、推进社会进步。

关于本次大会的定位，徐文伟也表示："华为认为作为一场旗舰大会应该拥有行业最前沿的内容，能吸引各类型、各层级的行业领袖和精英参会，成为企业高层商业与战略的交流平台。”

同时，徐文伟指出：“在这次HUAWEI CONNECT全联接大会上，华为将系统阐述华为业务战略方向、关键举措和实践，分享行业最前瞻的洞察，为行业领袖和精英提供一个商业与战略对话平台，全方位构建华为的行业领导力。华为将这次大会打造成一次生态大会，与客户、合作伙伴、产业组织、开发者共同引领并推动产业发展，发布创新领先的ICT产品与解决方案。”

**标签：华为 ,云计算 it168网站原创 ©2016-08-26**

图 13-54　HTML5 与 CSS3 多列应用页面

2．将【例 13-6-8】中 div 沿对角线运动改为沿矩形边框运动。要求沿每条边运动时改变背景颜色。

# 第 14 章　JavaScript 基础

**本章学习目标**

采用 HTML+CSS 技术设计的网页具有信息丰富、呈现样式美观等优势，但是网页还是缺乏与用户的交互功能。例如在网页中使用表单采集用户信息，需要对表单中输入的各类信息进行有效性验证，需要不断地向服务器端发送请求，一旦服务器响应不及时，导致网络访问中断，就有可能损失一个潜在的用户或一定的潜在市场份额，这对商家来说无疑是一大损失。JavaScript 是一种基于对象和事件驱动并具有相对安全性的客户端脚本语言，主要目的是为服务器端脚本语言提供数据验证的基本功能。

Web 前端开发工程师应掌握以下内容:

- 理解 JavaScript 程序的概念与作用。
- 掌握 JavaScript 标识符和变量的概念及其使用方法。
- 掌握 JavaScript 常用运算符和表达式概念。
- 掌握 JavaScript 中顺序、分支、循环三种程序控制结构语法。
- 掌握 JavaScript 函数的定义方法，并学会使用。
- 学会综合运用 JavaScript 设计具有动态、交互功能的网页。

JavaScript 是目前非常流行、应用广泛的一门客户端脚本语言，在 2017 年 11 月的 Tiobe（The Importance Of Being Earnest）编程语言排行榜中，JavaScript 排名第 6 位。JavaScript 是一种基于对象和事件驱动并具有相对安全性的客户端脚本语言，被广泛应用于各种客户端 Web 程序开发中，尤其是 HTML 的开发，能给 HTML 网页添加动态功能，响应用户各种操作，实现诸如信息验证、数字日历、跑马灯、显示浏览器停留时间等的特殊功能和效果。

## 14.1　JavaScript 概述

JavaScript 由 Netscape 公司的 Brendan Eich（布兰登•艾奇）于 1995 年开发设计，最初命名为 LiveScript，是一种动态、弱类型、基于原型的语言。后来 Netscape 与 Sun 公司进行合作，将 LiveScript 改名为 JavaScript。JavaScript 在设计之初受到 Java 的启发，语法上与 Java 有很多类似之处，并借用了许多 Java 的名称和命名规范。

### 14.1.1　JavaScript 简介

JavaScript 主要运行在客户端，用户访问带有 JavaScript 的网页，网页里的 JavaScript

程序就传给浏览器，由浏览器解释和处理。表单数据有效性验证等互动性功能，都是在客户端完成的，不需要和 Web 服务器发生任何数据交换，因此，不会增加 Web 服务器的负担。

JavaScript 具有如下特点。

**1．简单性**

JavaScript 是一种脚本编程语言，采用小程序段的方式实现编程，像其他脚本语言一样，JavaScript 是一种解释性语言，因此 JavaScript 编写的程序无须进行编译，而是在程序运行过程中被逐行地解释。JavaScript 基于 Java 基本语句和控制流，学习过 Java 的编程人员非常容易上手。此外它的变量类型采用弱类型，未使用严格的数据类型安全检查。

**2．安全性**

JavaScript 是一种安全性语言，它不允许程序访问本地的硬盘资源，不能将数据存入到服务器上，不允许对网络文档进行修改和删除，只能通过浏览器实现信息浏览或动态交互，从而有效地保障数据的安全性。

**3．动态性**

JavaScript 可以直接对用户的输入信息进行简单处理和响应，而无须向 Web 服务程序发送请求再等待响应。JavaScript 的响应采用事件驱动的方式进行，当页面中执行了某种操作时会产生特定事件（Event），例如移动鼠标、调整窗口大小等，会触发相应的事件响应处理程序。

**4．跨平台性**

JavaScript 程序运行只依赖于浏览器，与操作系统和机器硬件无关，只要机器上安装支持 JavaScript 的浏览器（例如 Internet Explorer、Firefox、Chrome 等）都能正确运行。

## 14.1.2 第一个 JavaScript 程序

JavaScript 程序不能独立运行，必须依赖于 HTML 文件。通常将 JavaScript 代码放置在 script 标记内，由浏览器 JavaScript 脚本引擎来解释执行。

**1．基本语法**

```
<script type="text/javascript" [src="外部js文件"]>
  js语句块;
</script>
```

**2．语法说明**

script 标记是成对标记，以<script>开始，以</script>结束。type 属性说明脚本的类型，属性值“text/javascript”意思是使用 JavaScript 编写的程序是文本文件。src 属性是可选属性，用于加载指定的外部 js 文件。如果设置此属性，将忽略 script 标记内的所有语句。

script 标记既可以放在 HTML 的头部，也可以放在 HTML 的主体部分，只是装载的时间不同。script 标记还有另一种说明格式，如下所示：

```
<script language="javascript"[src="外部js文件"]>…</script>
```

**【例 14-1-1】**使用 JavaScript 向 HTML 页面输出信息。代码如下所示，页面效果如图 14-1 所示。




图 14-1　第一个 JavaScript 实例



视频讲解

```
1 <!-- edu_14_1_1.html -->
2 <!doctype html>
3 <html lang="en">
4     <head>
5         <meta charset="UTF-8">
6         <title>第一个JavaScript实例</title>
7     </head>
8     <body>
9         <script type="text/javascript">
10            document.write("第一个JavaScript实例!");
11        </script>
12    </body>
13 </html>
```

**3．代码解释**

代码中第 9～11 行在 body 标记中直接插入 script 标记，第 10 行在 script 标记内利用 document.write()命令向页面写入“第一个 JavaScript 实例!”。

## 14.1.3　JavaScript 放置的位置

JavaScript 代码一般放置在页面的 head 或 body 部分。当页面载入时，会自动执行位于 body 部分的 JavaScript，【例 14-1-1】即是如此；而位于 head 部分的 JavaScript 只有被显式调用时才会被执行，如【例 14-1-2】所示。

**1．head 标记中的脚本**

script 标记放在头部 head 标记中，JavaScript 代码必须定义成函数形式，并在主体 body 标记内调用或通过事件触发。放在 head 标记内的脚本在页面装载时同时载入，这样在主体 body 标记内调用时可以直接执行，提高了脚本执行速度。

1）基本语法

```
function functionname(参数1,参数2,...,参数n){
    函数体语句;
}
```

2）语法说明

JavaScript 自定义函数必须以 function 关键字开始，然后给自定义函数命名，函数命名时一定遵守标识符命名规范。函数名称后面一定要有一对括号“()”，括号内可以有参数，也可以无参数，多个参数之间用逗号“,”分隔。函数体语句必须放在大括号“{}”内。

**【例 14-1-2】**在 head 标记内定义两个 JavaScript 函数。代码如下所示，页面效果如图 14-2 所示。

视频讲解

```
<!-- edu_14_1_2.html -->
<!doctype html>
<html lang="en">
    <head>
```

```
5        <meta charset="UTF-8">
6        <title>head中定义的JS函数</title>
7        <script type="text/javascript">
8           function message(){
9              alert("调用JS函数! sum(100,200)="+sum(100,200));  }
10          function sum(x,y){return x+y;//返回函数计算结果}
11       </script>
12    </head>
13    <body>
14       <h4>head标记内定义两个JS函数</h4>
15       <p>无返回值函数: message()</p>
16       <p>有返回值函数: sum(x,y)</p>
17       <form>
18          <input name="btnCallJS" type="button" onclick="message();"
            value="计算并显示两个数的和">
19       </form>
20    </body>
21 </html>
```

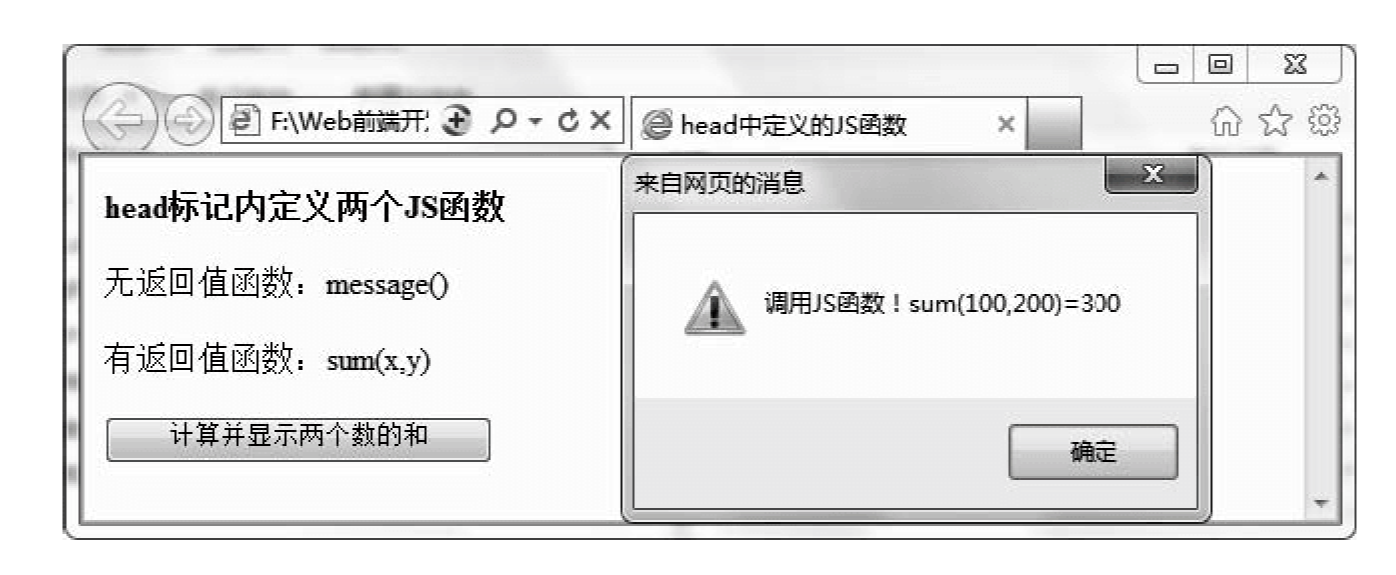



图 14-2　调用 head 标记中定义的 JavaScript 函数

3）代码解释

代码中第 7～11 行在 head 部分插入 script 标记，在 script 标记内定义 JavaScript 函数 message()、sum(x,y)。第 9 行用 alert 函数调用告警消息框，并调用 sum(100,200)函数，计算出结果并输出相关信息；第 18 行定义了一个普通按钮 btnCallJS，当单击该按钮时触发按钮的 onclick 事件，调用在 head 部分定义的 message 函数，弹出告警框。

**2．body 标记中的脚本**

script 标记放在主体 body 标记中，JavaScript 代码可以定义成函数形式，在主体 body 标记内调用或通过事件触发。也可以在 script 标记内直接编写脚本语句，在页面装载时同时执行相关代码，这些代码执行的结果直接构成网页的内容，在浏览器中可以查看，如【例 14-1-1】所示。

**3．外部 js 文件中的脚本**

除了将 JavaScript 代码写在 head 和 body 部分以外，也可将 JavaScript 函数单独写成一个 js 文件，在 HTML 文档中引用该 js 文件。

**【例 14-1-3】**调用外部 js 文件中的 JavaScript 函数。代码如下所示，页面效果如图 14-3 所示。



视频讲解

```
<!-- demo.js -->
function message()
{
   alert("调用外部js文件中的函数! ");
}
```




图 14-3　调用外部 js 文件的 JavaScript 函数

上述代码将 JavaScript 函数写在一个文件 demo.js 中，代码中第 2～5 行定义了一个函数 message()，注意在“.js”文件中不需要使用<script></script>标记来包围代码。

```
1 <!-- edu_14_1_3.html -->
2 <!doctype html>
3 <html lang="en">
4     <head>
5         <meta charset="UTF-8">
6         <title>调用外部js文件的JavaScript函数</title>
7         <script type="text/javascript" src="demo.js">
8             document.write("这条语句没有执行，被忽略掉了！");
9         </script>
10    </head>
11    <body>
12        <form>
13            <input name="btnCallJS" type="button" onclick="message()"
              value="调用外部js文件的JavaScript函数">
14        </form>
15    </body>
16 </html>
```

上述代码中第 7 行引用外部的 demo.js 文件；第 13 行定义普通按钮，在单击按钮时触发 onclick 事件，执行 demo.js 中定义的 message()函数实现在页面上弹出告警框的功能。很显然第 8 行代码没有被执行，因为设置 src 属性后，脚本<script></script>标记之间所有语句都不会执行，所以没有在页面上输入信息。

**4．事件处理的代码中的脚本**

JavaScript 代码除上述三种放置位置外，还可直接写在事件处理代码中。

**【例 14-1-4】**调用直接写在事件处理代码中 JavaScript 程序。代码如下所示，页面效果如图 14-4 所示。

视频讲解

```
<!-- edu_14_1_4.html -->
<!doctype html>
<html lang="en">
    <head>
        <meta charset="UTF-8">
        <title>直接在事件处理代码中加入JavaScript代码</title>
    </head>
    <body>
        <form>
            <input type="button" onclick="alert('直接在事件处理代码中加入
            JavaScript代码')" value="直接调用JavaScript代码">
        </form>
```

```
12     </body>
13 </html>
```

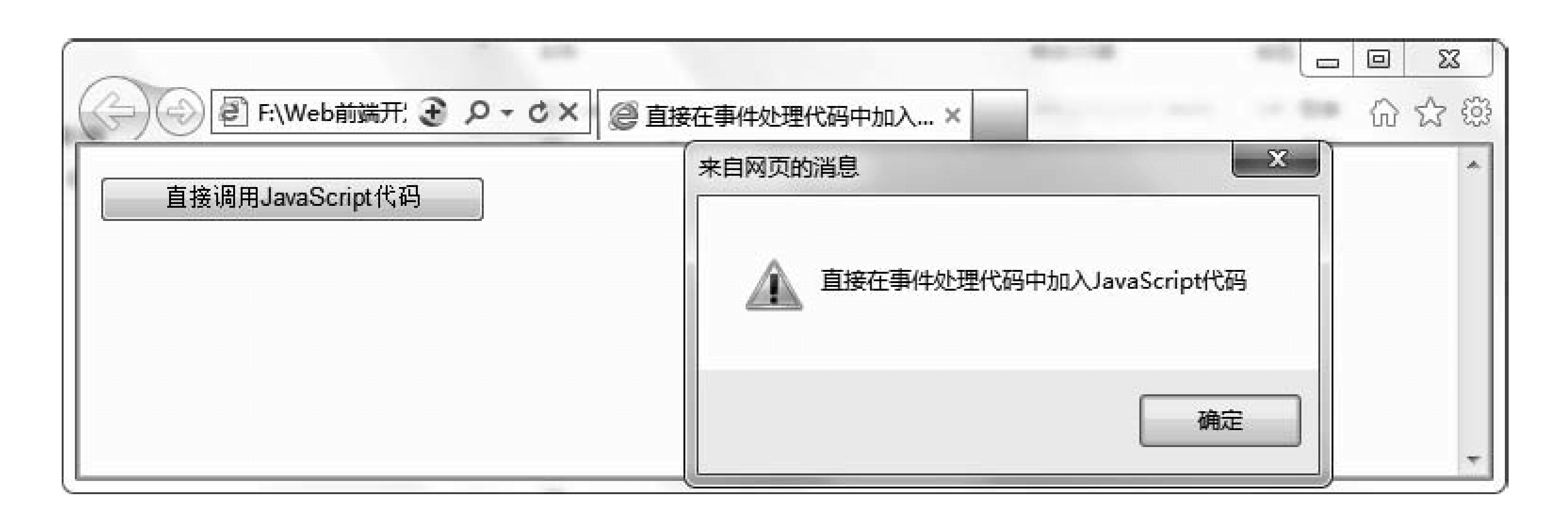



图 14-4　直接在事件处理代码中加入 JavaScript 代码

上述代码中第 10 行直接在普通按钮的 onclick 事件中插入了 JavaScript 代码，注意 JavaScript 代码需要用双引号（""）引起来，单击该按钮时弹出告警框。

用浏览器打开 JavaScript 程序时，安全级别设置较高的浏览器会阻止程序的运行，如图 14-5 所示。单击提示信息，弹出上下文菜单，选择“允许阻止的内容”选项，方可运行。

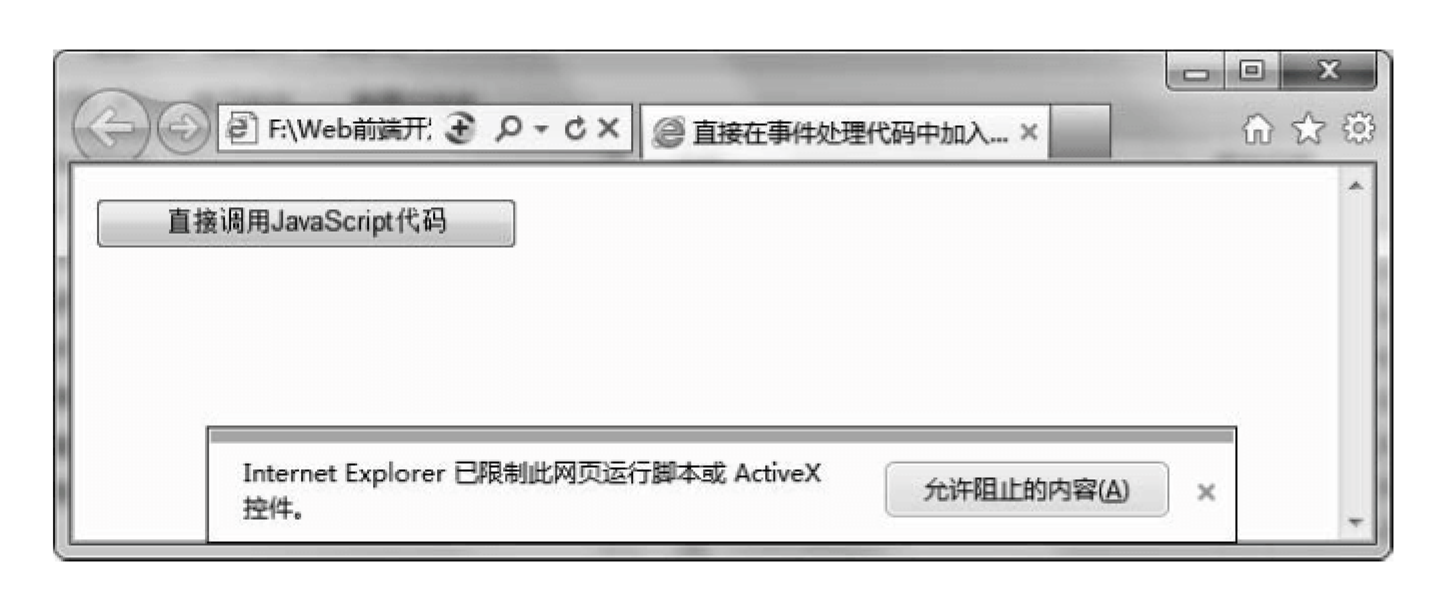



图 14-5　浏览器阻止程序运行界面

# 14.2　JavaScript 程序

JavaScript 程序由语句、语句块、函数、对象、方法、属性等构成，通过顺序、分支和循环三种基本程序控制结构来进行编程。

## 14.2.1　语句和语句块

JavaScript 语句向浏览器发出的命令。语句的作用是告诉浏览器该做什么。如下面语句的作用是告诉浏览器在页面上输出“我是 JavaScript 程序!”。

```
document.write("我是JavaScript程序! ");
```

多行 JavaScript 语句可以组合起来形成语句块，语句块以左大括号“{”开始，以右大括号“}”结束，块的作用是使语句序列一起执行。下面语句块向网页输出一个标题以及两个段落。

```
<script type="text/javascript">
{
   document.write("<h1>标题1</h1>");
```

```
    document.write("<p>这是段落1</p>");
    document.write("<p>这是段落2</p>");
}
</script>
```

### 14.2.2 代码

JavaScript 代码是 JavaScript 语句的序列，由若干条语句或语句块构成，以下代码中第 2～7 行由语句和语句块构成的就是 JavaScript 代码。

```
1 <script type="text/javascript">
2     var color="red";
3     if(color=="red")
4     {
5         document.write("颜色是红色!");
6         alert("颜色是红色!");
7     }
8 </script>
```

### 14.2.3 消息对话框

JavaScript 中的消息对话框分为告警框、确认框和提示框三种。

**1．告警框**

alert()函数用于显示带有一个图标、一条指定消息和一个“确定”按钮的告警框。

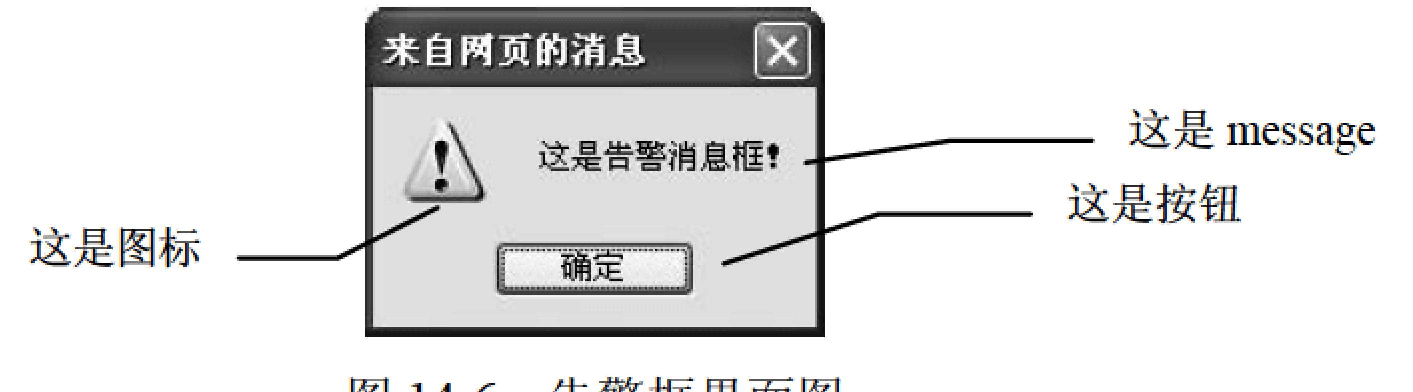
1）基本语法

```
alert(message);
```

2）参数说明

message 参数是显示在弹出对话框窗口上的纯文本（非 HTML 文本）。

**【例 14-2-1】**输出告警消息。代码如下所示，页面效果如图 14-6 所示。






视频讲解

图 14-6　告警框界面图

```
<!-- edu_14_2_1.html -->
<!doctype html>
<html lang="en">
    <head>
        <meta charset="UTF-8">
        <title>告警消息框的应用</title>
    </head>
    <body>
        <script type="text/javascript">
            alert("这是告警消息框! ");
        </script>
    </body>
</html>
```

3）代码解释

代码中第 10 行使用 alert()函数在页面弹出告警消息框。

**2．确认框**

confirm()方法用于显示带有一个图标、指定消息和“确定”及“取消”按钮的对话框。

1）基本语法

```
confirm(message);
```

2）语法说明

如果用户单击“确定”按钮，则 confirm()返回 true。如果单击“取消”按钮，则 confirm()返回 false。在用户单击“确定”按钮或“取消”按钮关闭对话框之前，它将阻止用户对浏览器的所有操作。在调用 confirm()时，将暂停对 JavaScript 代码的执行，在用户做出响应之前，不会执行下一条语句。

3）参数说明

message 参数是显示在弹出对话框窗口上的纯文本（非 HTML 文本）。

**【例 14-2-2】**使用 JavaScript 确认框。代码如下所示，页面效果如图 14-7 所示。

视频讲解

```
 1 <!-- edu_14_2_2.html -->
 2 <!doctype html>
 3 <html lang="en">
 4     <head>
 5         <meta charset="UTF-8">
 6         <title>确认框的应用</title>
 7         <script type="text/javascript">
 8             function show_confirm(){
 9                 var tf=confirm("请选择按钮!");
10                 if (tf==true){alert("您按了确定按钮！ ");    }
11                 else {alert("您按了取消按钮！ ");}
12             }
13         </script>
14     </head>
15     <body>
16         <form method="post" action="">
17             <input type="button" onclick="show_confirm()"value="显示确认框"/>
18         </form>
19     </body>
20 </html>
```




图 14-7　确认框使用界面图

4）代码解释

代码中第 8～12 行定义 JavaScript 函数 show_confirm()；第 9 行调用 confirm()函数显

示一个确认框；第 10～11 行使用双分支结构，如选择“确定”按钮则弹出告警框显示“您按了确定按钮!”，否则弹出告警框显示“您按了取消按钮!”。第 17 行在表单中插入一个按钮，并定义按钮的 onclick 事件，当用户单击按钮时调用 show_confirm()函数。

**3. 提示框**

prompt()方法用于提示用户在进入页面前输入某个值。

1）基本语法

```
prompt("提示信息",默认值);
```

如果用户单击提示框的取消按钮，则返回 null。如果用户单击确定按钮，则返回文本输入框中输入的值。在用户单击“确定”按钮或“取消”按钮关闭对话框之前，它将阻止用户对浏览器的所有操作。在调用 prompt()时，将暂停对 JavaScript 代码的执行，在用户做出响应之前，不会执行下一条语句。

2）参数说明

该函数有两个参数。第 1 个是“提示信息”；第 2 个是文本框的默认值，可以修改。

**【例 14-2-3】** 使用 JavaScript 提示框。代码如下所示，页面效果如图 14-8 所示。

视频讲解

```
1 <!-- edu_14_2_3.html -->
2 <!doctype html>
3 <html lang="en">
4     <head>
5         <meta charset="UTF-8">
6         <title>提示框的应用</title>
7         <script type="text/javascript">
8         function disp_prompt(){
9             var name=prompt("请输入您的姓名","李大为");
10            if (name!=null && name!="") //既不为空，也不为null
11            {
12                document.write("您好，" + name + "!");
13            }
14        }
15        </script>
16    </head>
17    <body>
18        <form method="post" action="">
19            <input type="button" onclick="disp_prompt()"value="显示提示框"/>
20        </form>
21    </body>
22 </html>
```

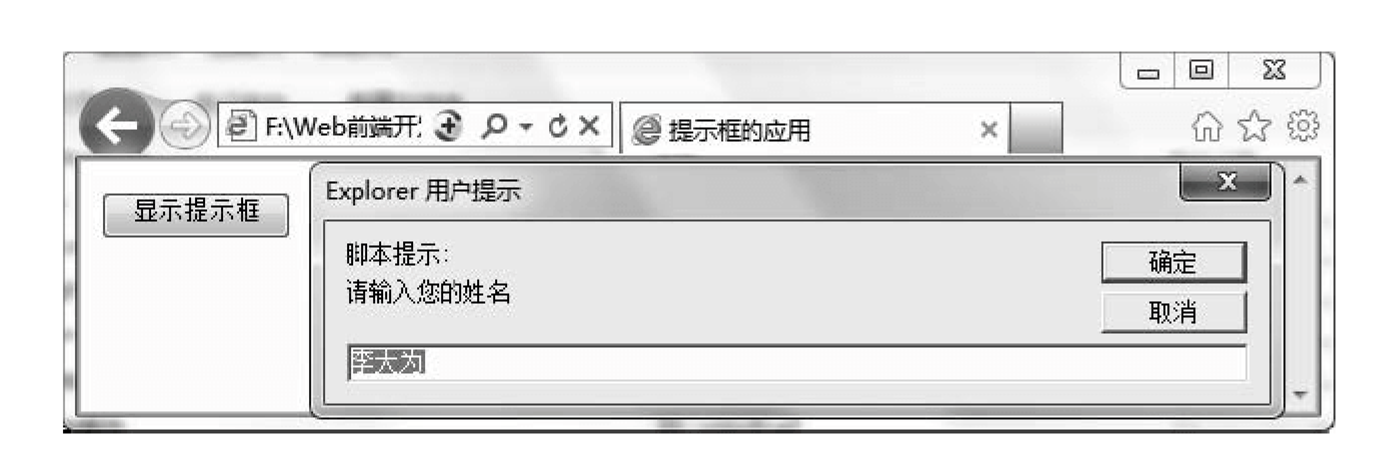





图 14-8　提示框使用界面图

3）代码解释

代码中第 8～14 行定义 JavaScript 函数 disp_prompt()；第 9 行使用 prompt()函数调用提

示框，让用户输入姓名；假设用户输入姓名为李大为，则第 12 行在页面输出信息“您好，李大为！”。

## 14.2.4　JavaScript 注释

视频讲解

JavaScript 提供了两种类型的注释：单行注释和多行注释。单行注释使用“//”作为注释标记，可以单独一行或跟在代码末尾，放在同一行中，“//”后为注释内容部分。注释行数较少时适宜使用单行注释，如果注释行数较多，则需要在每行的开头加“//”，比较麻烦，此时应使用多行注释。多行注释能包含任意行数的注释文本，以“/*”标记开始，以“*/”标记结束，两个标记之间所有的内容都是注释文本。所有注释的内容将被浏览器忽略，不影响页面效果和程序执行，对以后阅读和维护程序十分方便。

如果在某行代码前面加上单行注释“//”符号，那么此行代码就不能执行。对程序调试非常有用。例如：

```
<!-- edu_14_2_4.html -->
<!doctype html>
<html lang="en">
   <head>
      <meta charset="UTF-8">
      <title>注释的应用</title>
   </head>
   <body>
      <script type="text/javascript">
         //这是单行注释
         /*
            这是多行注释
            可以包含多行内容
         */
         //alert("此语句不执行！");
         alert("此语句执行了！");//执行时弹出告警消息框
      </script>
   </body>
</html>
```

# 14.3　标识符和变量

在任何一种编程语言中，实际编程时都要使用变量来存储常用的数据。所谓变量，顾名思义，就是在程序运行过程中不断变化的量。为了便于变量的使用，在使用时需要给变量命名，变量的名字则称为标识符。

## 14.3.1　命名规范

**1．标识符**

标识符是计算机语言中用来表示变量名、函数名等的有效字符序列，简单来说，标识符就是一个名字，JavaScript 关于标识符的规定如下：

（1）必须以英文字母或者下画线开头。

（2）必须由英文字母、数字、下画线组成，不能出现空格或制表符。

（3）不能使用 JavaScript 关键字与 JavaScript 保留字。

（4）不能使用 JavaScript 语言内部的单词，例如 Infinity、NaN、undefined 等。

（5）大小写敏感，如 name 和 Name 是不同的两个标识符。

根据以上规则，判断下列标识符命名是否是合法的。

```
合法的标识符：Hello_javascript、_12th、Dog119、$dcv
不合法的标识符：if、3Com、case、switch、break、class
```

**2．关键字**

关键字是 JavaScript 中已经被赋予特定意义的一些单词，关键字不能作为标识符来使用，JavaScript 中主要的关键字如表 14-1 所示。

**表 14-1　JavaScript 中主要的关键字**

| | | | | |
|---|---|---|---|---|
| break | case | catch | continue | default |
| delete | do | else | finally | for |
| function | if | in | instanceof | new |
| return | switch | this | throw | try |
| typeof | var | void | while | with |

**3．保留字**

JavaScript 中除了关键字以外，还有一些用于未来扩展时使用的保留字，保留字同样不能用于标识符的定义，JavaScript 中主要的保留字如表 14-2 所示。

**表 14-2　JavaScript 中主要的保留字**

| | | | | |
|---|---|---|---|---|
| abstract | boolean | byte | char | class |
| var | debugger | double | enum | export |
| extends | final | float | goto | implements |
| import | int | interface | long | native |
| package | private | protected | public | short |
| static | super | synchronized | throws | transient |
| volatile | | | | |

## 14.3.2　数据类型

数据类型是每一种计算机语言的重要基础，JavaScript 中的数据类型可分为字符型、数值型、布尔型、Null、Undefined 和对象六种类型。

**1．String 字符型**

字符型数据又称为字符串，由若干个字符组成，并且需要用单引号（''）或双引号（""）封装起来（在 JavaScript 中，使用单引号和双引号的效果是一样的），下面的例子列举了正确和错误使用字符型数据的两种情形：

```
"Tiger", 'JavaScript字符串'          （正确）
'document", "你好'                   （错误，单引号双引号不匹配）
```

在使用字符串的过程中，有时会遇到一种情况：在一个字符串中需要使用单引号或双

引号。正确的方法是在由双引号标记的字符串中加入引用字符时使用单引号，在由单引号标记的字符串中加入引用字符时使用双引号，即保证一个字符串的开头和结尾使用同一种引号，而字符串内使用另一种引号。下面给出了正确的用法：

```
"热烈欢迎参加'JavaScript技术'研讨的专家"
```

**2．Number 数值型**

与其他编程语言类似，JavaScript 中最基本的数据类型之一是数值型，该类型可分为整型、浮点型、内部常量以及特殊值。

（1）整型：例如 100、-3500、0 等都是整数。整数除了以十进制表示外，还可以八进制和十六进制的方式表示。使用 0 开头的整数是八进制整数，如 017、-035 等都是合法的八进制整数。使用 0x 或 0X 开头的整数是十六进制整数，如 0x16、0X3A89 等都是合法的十六进制整数。

（2）浮点型：例如 3.53、-534.87 等都是浮点型数值。浮点数还可以采用科学计数法进行表示，如 3.5E15 表示 $3.5\times10^{15}$。

（3）内部常量：JavaScript 中常用的内部常量及说明如表 14-3 所示。

**表 14-3　JavaScript 中的内部常量及说明**

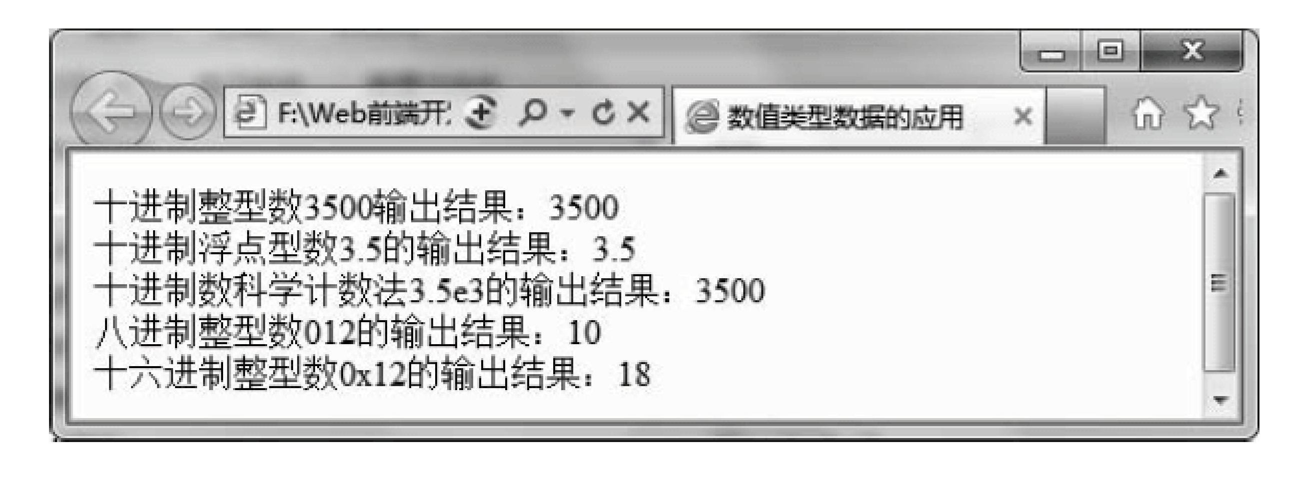
| 常　　量 | 说　　明 | 常　　量 | 说　　明 |
|---|---|---|---|
| Math.E | 自然数 | Math.LN2 | 2 的自然对数 |
| Math.PI | 圆周率 | Math.LN10 | 10 的自然对数 |
| Math.SQRT2 | 2 的平方根 | Math.LOG2E | 以 2 为底的 e 的对数 |
| Math.SQRT1_2 | 1/2 的平方根 | Math.LOG10E | 以 10 为底的 e 的对数 |

（4）特殊值：JavaScript 中的特殊值及说明如表 14-4 所示。

**表 14-4　JavaScript 中的特殊值及说明**

| 特　殊　值 | 说　　明 |
|---|---|
| Infinity | 无穷大 |
| Number.NaN | 非数字值（Not a Number） |
| Number.MAX_VALUE | 可表示的最大的数 |
| Number.MIN_VALUE | 可表示的最小的数 |
| Number.NEGITIVE_INFINITY | 负无穷大，溢出时返回该值 |
| Number.POSITIVE_INFINITY | 正无穷大，溢出时返回该值 |

**【例 14-3-1】**数值类型数据的应用。代码如下所示，页面效果如图 14-9 所示。





视频讲解

图 14-9　数值类型数据使用实例

```
1 <!-- edu_14_3_1.html -->
2 <!doctype html>
3 <html lang="en">
4     <head>
5         <meta charset="UTF-8">
6         <title>数值类型数据的应用</title>
7     </head>
8     <body>
9         <script type="text/javascript">
10            var i = 3500,f = 3.5,s = 3.5e3;
11            var o = 012,h = 0x12;
12            document.write("十进制整型数"+i+"的输出结果："+i+"<br>");
13            document.write("十进制浮点型数"+f+"的输出结果："+f+"<br>");
14            document.write("十进制数科学计数法3.5e3的输出结果："+s+"<br>");
15            document.write("八进制整型数012的输出结果："+o+"<br>");
16            document.write("十六进制整型数0x12的输出结果："+h+"<br>");
17        </script>
18    </body>
19 </html>
```

上述代码中第 10 行定义变量 i 是整数、变量 f 是浮点数、变量 s 是浮点数，并用科学计数法表示，相当于 $3.5\times10^3$，即 3500；第 11 行定义变量 o 是一个八进制数，相当于十进制的 10；同时定义变量 h 是一个十六进制数，相当于十进制数的 18。

**3．Boolean 布尔型**

布尔型是一种只含有 true 和 false 这两个值的数据类型，通常来说，布尔型数据表示“真”或“假”。在实际应用中，布尔型数据常用在比较、逻辑等运算中，运算的结果往往就是 true 或者 false。例如 1<2 的比较结果是 true，而 3==4 的比较结果是 false。此外，布尔型变量还常用在控制结构的语句中，如 if 语句等。

JavaScript 中，通常采用 true 和 false 表示布尔型数据，但也可将他们转换为其他类型的数据，例如可将值为 true 的布尔型数据转换为整数 1，而将值为 false 的布尔型数据转换为整数 0。但不能用 true 表示 1 或 false 表示 0。

**4．Null**

在 JavaScript 中，Null 是一种特殊的数据类型，也称为空类型，此类型只有一个值为 null，表示“无值”，什么也不表示。null 除了表示 Null 类型的数据外，也可用在表示其他类型的数据中，例如对象、数组和字符串等。当变量不再使用时，将它赋值为 null，以释放存储空间。

**5．Undefined**

在 JavaScript 中，Undefined 也是一类特殊的值，是指变量创建之后还没有赋值之前所具有的值，则返回值就是 Undefined。它与 null 值的不同之处在于：null 值表示已经对变量赋值，只不过赋的值是“无值”；而 Undefined 表示变量不存在或者没有赋值。如果使用未定义的变量也会显示 undefined，但通常使用未定义的变量会造成程序错误。

**6．Object 对象**

在 JavaScript 中除了数值型、字符型和布尔型等这些基本的数据类型以外，还有一种复合的数据类型称为对象，对象是属性和方法的集合。对象的属性可以是任何类型的数据，包括数值、字符、布尔型，甚至是另一种类型的对象，而方法是一个定义在对象中的函数，

用于实现特定的功能。

JavaScript 中定义了多个对象，如 Date、window、document 等，这部分内容将在第 16 章详细介绍。

## 14.3.3 变量

JavaScript 变量是一个存储或者表示数据的名称，可用来存储和表示各种数据类型的数据，并且这些值在程序运行期间是可以改变的。JavaScript 是一种无数据类型的计算机语言，在定义变量时不需要指定变量的数据类型，统一使用关键字 var 声明，JavaScript 会在需要的时候自动对不同的数据类型进行转型。

**1．基本语法**

```
var 变量名[=初值][,变量名[=初值]…] ;
```

**2．语法说明**

var（variant）是关键字，声明时至少要有一个变量，并给每个变量命名。

变量命名应该符合标识符命名规范。

可以同时声明多个变量，多个变量之间用逗号“,”分隔。

可以边声明边赋值。

每条声明语句均需要以“;”结束，这是一个好习惯。

以下是声明变量的示例。

```
var userName, userAge;
var str;
```

上面例子中第 1 行代码声明了两个变量，第 2 行声明了一个变量 str。

```
var x1= 0, y1= 2.5;          //声明时同时赋值
var str = "欢迎学习JS";       //声明时同时赋值
```

在 JavaScript 中，所使用的变量也可以不声明直接使用，但这不是一种好的编程习惯，建议所有变量“先声明再使用”。

```
变量名=初值;
```

例如：

```
x1 = 0, y1 = 2.5;
str = "欢迎学习JS";
userEmail="jlchu@163.com";      //向未声明的变量直接赋值
```

代码中第 1 行给 x1 赋值为 0，y1 赋值为 2.5，第 2 行给 str 赋值为“欢迎学习 JS”。在实际使用中也可将变量的声明和赋值合并成一条语句书写。第 3 行直接向未声明变量赋值。

## 14.3.4 转义字符

如果在字符串中涉及一些特殊字符，如“\”、“"”、“'”等，这些字符无法直接使用，需要采用转义字符的方式。JavaScript 中常用的转义字符如表 14-5 所示。

表 14-5　常用转义字符

| 转 义 字 符 | 代 表 含 义 | 转 义 字 符 | 代 表 含 义 |
| --- | --- | --- | --- |
| \b | 退格符 | \t | 水平制表符 |
| \f | 换页符 | \' | 单引号 |
| \n | 换行符 | \" | 双引号 |
| \r | 回车符 | \\ | 反斜线 |
| \uhhhh | 编码转换 | | |

# 14.4　运算符和表达式

JavaScript 运算符主要有算术运算符、关系运算符、逻辑运算符、赋值运算符、自增、自减运算符、条件运算符、逗号运算符和位运算符等；也可以根据操作数的个数，将运算符分为一元运算符、二元运算符和三元运算符。

由操作数（变量、常量、函数等）和运算符结合在一起构成的式子称为“表达式”，最简单的表达式可以是常量名称，例如以下都是合法的表达式：

```
100                         //整型常量表达式
14.35                       //浮点型常量表达式
"JavaScript"                //字符型常量表达式
x                           //变量表达式
```

此外，还可以使用操作数和运算符建立复杂的表达式，例如“str="江苏省";”是一个赋值表达式，将字符串"江苏省"赋值给变量 str，还有其他类型的表达式将在下面详细介绍。

## 14.4.1　算术运算符和表达式

JavaScript 算术运算符负责算术运算，用算术运算符和运算对象（操作数）连接起来符合规则的式子，称为算术表达式。JavaScript 中常用算术运算符如表 14-6 所示。

表 14-6　算术运算符

| 运　算　符 | 操 作 说 明 | 运　算　符 | 操 作 说 明 |
| --- | --- | --- | --- |
| + | 加法运算符 | % | 模（取余）运算符 |
| – | 减法运算符或取反运算符 | ++ | 自增运算符 |
| * | 乘法运算符 | -- | 自减运算符 |
| / | 除法运算符 | | |

**1．基本语法**

二元运算符：

```
op1 operator op2
```

一元运算符：

```
op operator 或 operator op
```

**2．语法说明**

算术运算符是一类常见的、较为熟悉的运算符，但作为一种编程语言，也有一些需要

特别注意的地方。

1）加法运算符（+）

加法运算符是一个二元运算符，可以对数值型的操作数执行加法操作。例如：

```
304+135;                //对数字304和135执行加法操作，结果为439
```

此外，加法运算符还可以用在其他情况中。如果两个操作数都是字符型，或者一个是字符型、另一个是数值型，那么加法运算将数值转换成字符串，然后执行两个字符串的连接操作。例如：

```
"Hello"+"JavaScript"; //对两个字符串执行连接操作，结果为"HelloJavaScript"
"JavaScript"+1.6   //将数值转换为字符，再与字符串连接操作，结果为"JavaScript1.6"
```

2）减法运算符（–）

减法运算符是一个二元运算符，对两个数值型操作数执行减法操作。例如：

```
888-303;                //对数字888和303执行减法操作，结果为585
```

如果减法运算符用于取反运算，那么它就是一个一元运算符，操作数必须为数字，且运算符位于操作数前。

```
-108;                   //操作数为108，取反结果为-108
-(-350);                //操作数为-350，取反结果为350
```

减法运算符还有一个作用，就是可以将字符串转换成数值型数据。例如：

```
"690"-0;                //将字符串"690"转换成数字690
```

3）乘法运算符（*）

乘法运算符是一个二元运算符，完成两个数值型操作数的乘法操作。如果操作数不是数字型，但可以转换为数值型，乘法运算符会自动将其转换为数字，再进行乘法操作；如果操作数无法转换成数值型，则运算结果为“NaN”。

```
3*5;                    //对数字3和5执行乘法操作，结果为15
3*"6";                  //将字符"6"转换为数字6，再执行乘法操作，结果为18
3*"A" ;                 //"A"无法转换为数字，结果为NaN
```

4）除法运算符（/）

除法运算符是一个二元运算符，完成两个数值型操作数的除法操作。其运算规则与乘法运算类似，如果操作数不是数字型，但可以转换为数值型，除法运算符会自动将其转换为数字，再进行除法运算；如果操作数无法转换成数值型，则运算结果为 NaN。如果被除数为正数，除数为 0，则结果为 Infinity；如果被除数为负数，除数为 0，则结果为-Infinity。

```
15/5;                   //对数字15和5执行除法操作，结果为3
18/"6" ;                //将字符"6"转换为数字6，再执行除法操作，结果为3
18/"A" ;                //"A"无法转换为数字，结果为NaN
20/0  ;                 //被除数为20，除数为0，结果为Infinity
-20/0  ;                //被除数为-20，除数为0，结果为-Infinity
```

5）模运算符（%）

模运算符又称取余数运算符，可以计算第一个操作数对第二个操作数的模（余数）。

模运算符同样可以将能够转换为数值型的操作数转换为数值型数据再运算，如果操作数无法转换为数值型，则取模结果为 NaN。另外，任何数字对 0 取模结果都是 NaN。

```
15 % 6;                //对数字15和6执行取模操作，结果为3
18 % "7" ;             //将字符"7"转换为数字7，再执行取模操作，结果为4
18 % "A" ;             //"A"无法转换为数字，结果为NaN
20 % 0 ;               //第二个操作数为0，结果为NaN
```

6）自增运算符（++）

自增运算符是一元运算符，可以对操作数执行自增运算，增量为 1。要求操作数必须是变量，不能是常量。自增运算有两种形式：前置和后置。前置是将自增运算符放在操作数之前，表示在使用操作数之前，先将其增加 1；后置是将自增运算符放在操作数之后，表示在使用完操作数之后，再将之增加 1。例如：

```
var x,y,a=3,b=5;
x=a++;                 //自增后置,x的值为3,a的值为4
y=++b;                 //自增前置,y的值为6,b的值为6
```

7）自减运算符（--）

自减运算符是一元运算符，可以对操作数执行自减运算，减量为 1。同样地，自减运算符也要求操作数必须是变量，不能是常量。自减运算有两种形式：前置和后置。前置是将自减运算符放在操作数之前，表示在使用操作数之前，先将其减少 1；后置是将自减运算符放在操作数之后，表示在使用完操作数之后，再将之减少 1。例如：

```
var x,y,a=8,b=10;
x=a--;                 //自减后置，x的值为8，a的值为7
y=--b;                 //自减前置，y的值为9，b的值为9
```

## 14.4.2　关系运算符和表达式

关系运算符用于比较运算符两端的表达式的值，确定二者的关系，根据运算结果返回一个布尔值。用关系运算符和操作数连接起来符合规则的式子，称为关系表达式。JavaScript 中常用关系运算符如表 14-7 所示。

表 14-7　关系运算符

| 运　算　符 | 操 作 说 明 | 运　算　符 | 操 作 说 明 |
|---|---|---|---|
| == | 等于 | <= | 小于等于 |
| != | 不等于 | > | 大于 |
| < | 小于 | >= | 大于等于 |
| === | 全等于 | !== | 非全等于 |

**1．基本语法**

```
op1 operator op2
```

**2．语法说明**

1）等于运算符（==）

等于运算符是一个二元运算符，用于判断两个操作数是否相等，如果相等返回 true，

如果不相等返回 false。有三点需要注意：

（1）操作数的类型转型。如果被比较的操作数是同类型的，那么等于运算符将直接对操作数进行比较。如果被比较的操作数类型不同，那么等于运算符在比较两个操作数之前会自动对其进行类型转换。转换规则为：

- 如果操作数中既有数字又有字符串，那么 JavaScript 将字符串转换为数字，然后进行比较。
- 如果操作数中有布尔型值，那么 JavaScript 将 true 转换为 1，将 false 转换为 0， 然后进行比较。
- 如果操作数一个是对象，一个是字符串或数字，那么 JavaScript 将把对象转换成与另一个操作数类型相同的值，然后再进行比较。

（2）两个对象、数组或者函数的比较不同于有字符串、数字和布尔值参与的比较。前者比较的是引用内容，换句话说，只有两个变量引用的是同一个对象、数组或者函数的时候，它们才是相等的；如果两个变量引用的不是同一个对象、数组和函数，即使它们的属性、元素完全相同，或者可以转换成相等的原始数据类型的值，它们也是不相等的。

（3）特殊值的比较。

- 如果一个操作数是 NaN，另一个操作数是数字或 NaN，那么结果是不等。
- 如果两个操作数都是 null，那么结果相等。
- 如果两个操作数都是 undefined 类型，那么结果相等。
- 如果一个操作数是 null，一个操作数是 undefined 类型，那么结果相等。

2）不等于运算符（!=）

不等于运算符和等于运算符的比较规则正好相反：如果两个操作数相等，则返回 false；如果两个操作数不等，则返回 true。除此之外，不等于运算符的数据类型转换规则，对象、数组和函数的比较方法，以及特殊值的处理情况都可以参考等于运算符的情况，等于运算符返回 true 时，不等于运算符返回 false；等于运算符返回 false 时，不等于运算符返回 true。

3）小于运算符（<）

小于运算符用于比较两个操作数，如果第一个操作数小于第二个操作数，那么计算结果返回 true，否则返回 false。

小于运算符存在数据类型转换问题，其规则是：

- 运算符可以是任何类型的，但是比较运算只能在数字和字符上执行，所以不是数字和字符类型的数据都会被转换成这两种类型。
- 如果两个操作数都是数字，或者都能被转换为数字，则按照数字大小规则比较。
- 如果两个操作数都是字符串，或者都能被转换为字符串，则按照字母顺序规则比较。
- 如果一个是字符串或者能被转换为字符串，一个是数字或者能被转换为数字，则首先将字符串转换成数字，然后按数字大小规则比较。
- 如果操作数中包含无法转换成数字也无法转换成字符串的内容，比较结果是 false。

4）小于等于运算符（<=）

小于等于运算符用于比较两个操作数，如果第一个操作数小于或等于第二个操作数，

那么计算结果返回 true，否则返回 false。数据类型转换规则参考小于运算符。

5）大于运算符（>）

大于运算符用于比较两个操作数，如果第一个操作数大于第二个操作数，那么计算结果返回 true，否则返回 false。数据类型转换规则参考小于运算符。

6）大于等于运算符（>=）

大于等于运算符用于比较两个操作数，如果第一个操作数大于或等于第二个操作数，那么计算结果返回 true，否则返回 false。数据类型转换规则参考小于运算符。

7）全等于号(===)与非全等于号(!==)

全等于号“===”表示比较的两个数据的值和类型均相等，结果为 true，若只是值相同，但类型不同，则结果为 false。例“9999”===9999，值相同类型不同，结果为 false。非全等于号“!==”表示比较的两个数值的值和类型有一个不相等，或两个都不相等。“9999”!==9999，值相同类型不同，结果为 true。

### 14.4.3　逻辑运算符和表达式

逻辑运算符用来执行逻辑运算，其操作数都应该是布尔型数值和表达式或者是可以转换为布尔型的数值和表达式，其运算结果返回 true 或 false。用逻辑运算符和操作数连接起来符合规则的式子，称为逻辑表达式。JavaScript 中常用逻辑运算符如表 14-8 所示。

**1．基本语法**

二元运算符：

```
boolean_expression1 operator boolean_expression2
```

一元运算符：

```
!boolean_expression
```

**2．语法说明**

1）逻辑与运算符（&&）

逻辑与运算符是一个二元运算符，如果两个布尔型操作数都是 true，则运算结果为 true；如果两个操作数中有一个或两个为 false，则运算结果返回 false。

```
true && false              //逻辑与运算结果为false
(8<10) && (3>-1)           //(8<10)为true，(3>-1)为true，逻辑与运算结果为true
```

**表 14-8　逻辑运算符**

| a | b | !a(逻辑非) | a&&b(逻辑与) | a\|\|b(逻辑或) |
|---|---|---|---|---|
| true | true | false | true | true |
| true | false | false | false | true |
| false | true | true | false | true |
| false | false | true | false | false |

2）逻辑或运算符（||）

逻辑或运算符是一个二元运算符，如果两个布尔型操作数中有一个或两个为 true，则运算结果返回 true；如果两个布尔型操作数全部为 false，则运算结果返回 false。

```
true || false ;          //逻辑或运算结果为true
(3>=5) || (2>0) ;        //(3>=5)为false，(2>0)为true，逻辑或运算结果为true
```

3）逻辑非运算符（!）

逻辑非运算符是一个一元运算符，其作用是先计算操作数的布尔值，然后对运算结果的布尔值取反，并作为结果返回，即如果操作数的布尔值为 true，则逻辑非的运算结果返回 false；反之运算结果返回 true。

```
!10   ;             //10先转换为布尔型变量true，逻辑非运算结果为false
!((4<10)&&(5>6)) ;//(4<10)为true，5>6为false，逻辑与运算结果为false，再进行逻辑非运算结果为true
```

### 14.4.4 赋值运算符和表达式

赋值运算符是 JavaScript 中使用频率最高的运算符之一。赋值运算符要求其左操作数是一个变量、数组元素或对象属性，右操作数是一个任意类型的值，可以为常量、变量、数组元素或对象属性。赋值运算符的作用就是将右操作数的值赋给左操作数。用赋值运算符和操作数连接起来符合规则的式子，称为赋值表达式。JavaScript 中常用赋值运算符列在表 14-9 中。

**表 14-9 赋值运算符**

| 运 算 符 | 操 作 说 明 | 运 算 符 | 操 作 说 明 |
|---|---|---|---|
| = | 基本赋值运算符 | *= | 复合赋值运算符，a*=b 同于 a=a*b |
| += | 复合赋值运算符，a+=b 同于 a=a+b | /= | 复合赋值运算符，a/=b 同于 a=a/b |
| –= | 复合赋值运算符，a–=b 同于 a=a-b | %= | 复合赋值运算符，a%=b 同于 a=a%b |

**1．基本语法**

简单赋值运算：

```
<变量>=<变量> operator <表达式>
```

复合赋值运算：

```
<变量> operator =<表达式>
```

**2．语法说明**

赋值运算符可以将一个值赋给一个变量名。

```
var a=10, b=20, c;
c = a;                  //c的值为10
c += b;                 //相当于c=c+b，c的值为30
c /= a;                 //相当于c=c/a，c的值为3
b %= c;                 //相当于b=b%c，b的值为2
```

### 14.4.5 位运算符和表达式

位运算符是对二进制表示的整数进行按位操作的运算符。如果操作数是十进制或者其他进制表示的整数，运算前先将这些整数转换成 32 位的二进制数字，如果操作数无法转换成 32 位的二进制数表示，位运算的结果为 NaN。

#### 1．基本语法

二元运算符：

```
op1 operator op2
```

一元运算符：

```
operator op
```

#### 2．语法说明

位运算符是在数的二进制基础上进行操作，JavaScript 中常用位运算符如表 14-10 所示。

表 14-10　位运算符

| 运　算　符 | 操作说明 | 运　算　符 | 操作说明 |
|---|---|---|---|
| & | 按位与运算符 | ~ | 按位非运算符 |
| \| | 按位或运算符 | ^ | 按位异或运算符 |

1）按位与运算符（&）

按位与运算符是一个二元运算符，它将两个整数型操作数转换为二进制数并逐位进行逻辑与操作。如果两个操作数对应位置上的数字都是 1，运算结果的这一位为 1，否则为 0。

```
10 & 78      //将十进制数10转换为二进制数00001010，将十进制数78转换为二进制数01001110，按位与结果为00001010，转换为十进制数为10
030 & 071    //将八进制数30转换为二进制数00011000，将八进制数71转换为二进制数00111001，按位与结果为00011000，转换为十进制数为24
```

2）按位或运算符（|）

按位或运算符是一个二元运算符，它将两个整数型操作数转换为二进制数并逐位进行逻辑或操作。如果两个操作数对应位置上的数字都是 0，运算结果的这一位为 0，否则为 1。

```
81 | 16  //将十进制数81转换为二进制数01010001，将十进制数16转换为二进制数00010000，按位或结果为01010001，转换为十进制数为81
0xA1 | 0x39  //将十六进制数A1转换为二进制数10100001，将十六进制数39转换为二进制数00111001，按位或结果为10111001，转换为十进制数为185
```

3）按位非运算符（~）

按位非运算符是一个一元运算符，其作用是将整型操作数转换为二进制数并逐位进行逻辑非操作，即将操作数的每一位取反（将 1 变为 0，0 变为 1）。

```
~100    //将十进制数100转换为二进制数00000000 00000000 00000000 01100100，按位
        非的结果为11111111 11111111 11111111 10011011（这是负数的补码），转换为十
        进制数为-101
~0xCD   //将十六进制数CD转换为二进制数00000000 00000000 00000000 11001101，按位
        非的结果为1111111 11111111 11111111 00110010（这是负数的补码），转换为十
        进制数为-206
```

4）按位异或运算符（^）

按位异或运算符是一个二元运算符，它将两个整数型操作数转换为二进制数并逐位进行逻辑异或操作。如果两个操作数对应位置上的数字相同（都为 0 或都为 1），运算结果的这一位为 0，否则为 1。例如：

```
10 ^ 30  //将十进制数10转换为二进制数00001010，将十进制数30转换为二进制数00011110，
          按位异或的结果为00010100，转换为十进制为20
0xA0 ^ 032//将十六进制数A0转换为二进制数10100000，将八进制数32转换为二进制位00011010，
            按位异或的结果为10111010，转换为十进制数为186
```

## 14.4.6 条件运算符和表达式

条件运算符是一个三元运算符，条件表达式由？、：运算符和三个操作数构成。

### 1．基本语法

```
<变量>=<条件表达式> ? <真值表达式> : <假值表达式>
```

### 2．语法说明

该条件表达式表示，如果条件表达式的结果为真（true），则将真值表达式的值赋给变量，否则将假值表达式的值赋给变量。

例如，变量 number1、number2 比较大小，将较大的数赋值给变量 max，代码如下：

```
var max=(number1>number2)?number1:number2;
```

## 14.4.7 其他运算符和表达式

JavaScript 中除了上述运算符外，还有一些其他运算符，如表 14-11 所示。

表 14-11 其他运算符

| 运算符 | 操作说明 | 运算符 | 操作说明 |
|---|---|---|---|
| , | 逗号运算符 | delete | 删除运算符 |
| new | 新建对象运算符 | typeof | 类型运算符 |

### 1．逗号运算符（,）

逗号运算符是一个二元运算符，其运算规则是先计算第一个表达式的值，再计算第二个表达式的值，运算结果为第二个表达式的值。

```
var rs;
rs = (3+5, 10*6);  //先计算第一个表达式3+5的值为8，再计算第二个表达式10×6的值为
                     60，最后将第二个表达式的值60赋给变量rs
```

### 2．新建对象运算符（new）

新建对象运算符是一个一元运算符，用于创建 JavaScript 对象实例或数组。

```
var obj = new Object();      //创建一个Object对象，对象名为obj
var date = new Date();       //创建一个Date对象，对象名为date
var array = new Array();     //创建一个数组对象，对象名为array
```

### 3．删除运算符（delete）

删除运算符是一个一元运算符，用于删除一个对象的属性或某个数组的元素。

```
delete array[30];            //删除array数组中下标为30元素（第31个元素）
delete obj.height;           //删除对象obj的height属性
```

### 4．类型运算符（typeof）

类型运算符是一个一元运算符，其操作数可以是任意类型，运算结果返回一个表示操作数类型的字符串。

```
typeof(300);                //运算结果为number
typeof("Hello");            //运算结果为String
```

typeof 运算符的具体规则如表 14-12 所示。

**表 14-12　类型运算符运算规则**

| 数 据 类 型 | 运 算 结 果 | 数 据 类 型 | 运 算 结 果 |
|---|---|---|---|
| 数字型 | Number | 数组 | Object |
| 字符型 | String | 函数 | Function |
| 布尔型 | Boolean | Null | Object |
| 对象 | Object | 未定义 | undefined |

# 14.5　JavaScript 程序控制结构

在 HTML 基础上，使用 JavaScript 可以开发交互式 Web 页面，如可以在线填写各类表格、联机编写文档并发布等。JavaScript 的出现使得网页和用户之间实现了一种实时性的、动态的、交互性的关系，使网页包含更多活跃元素和更加精彩的内容。这也是 JavaScript 与 HTML DOM 共同构成 Web 网页的行为。在网页设计中 JavaScript 的主要作用是实现内容与行为的分离，而要设计交互式的页面必须编写相应的脚本程序。程序是专门用来解决某一问题的特定代码。

JavaScript 程序设计分为面向过程和面向对象的程序设计。在所有的编程语言中，程序的结构都有三种，分别为顺序结构、分支结构和循环结构，任何复杂的算法均可以使用这三种结构来表达。

## 14.5.1　顺序结构

顺序结构是程序设计中最常用、最基本的一种程序结构，是按照语句出现的顺序，从第一条语句开始一步一步逐条执行，直至最后一条语句。

**【例 14-5-1】**使用顺序结构程序计算圆的周长和面积。代码如下所示，页面效果如图 14-10 所示。

视频讲解

```
 1 <!-- edu_14_5_1.html -->
 2 <!doctype html>
 3 <html lang="en">
 4     <head>
 5         <meta charset="UTF-8">
 6         <title>顺序结构的应用</title>
 7     </head>
 8     <body>
 9         <script type="text/javascript">
10             var radius = 6;
11             var circumference = 2 * Math.PI * radius;
12             var area = Math.PI * radius * radius;
13             alert("圆的周长为" + circumference+"\n"+"圆的面积为" + area);
14         </script>
15     </body>
16 </html>
```

**代码解释**

代码中使用了顺序结构计算圆的周长和面积。第 10 行定义圆的半径变量并赋值为 6；

第 11 行定义圆的周长变量并赋值；第 12 行定义圆的面积并赋值；第 13 行输出圆的周长和面积信息。整个代码从始至终都是按照代码书写的顺序一行一行执行，直到最后一条语句。

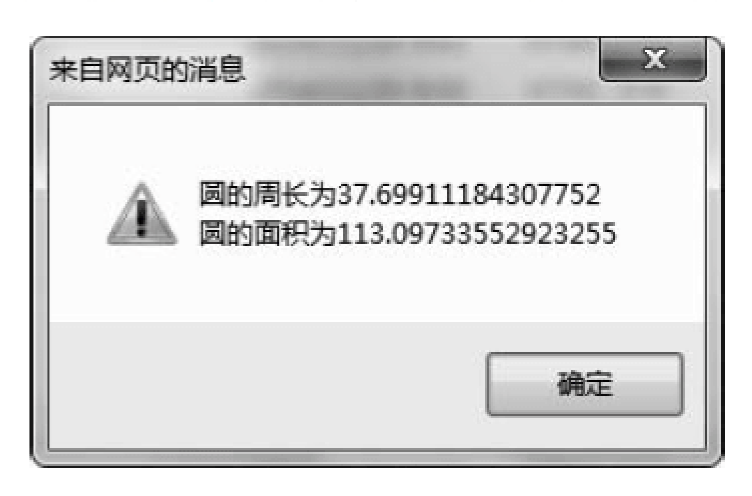



图 14-10　计算圆的周长和面积

## 14.5.2　分支结构

在 JavaScript 中可以使用四种形式分支结构语句：

- 单 if(){语句块}语句，在条件成立时执行语句块。
- 双 if(){语句块 1}else{语句块 2}语句，在指定条件成立时执行语句块 1，不成立时执行语句块 2。
- 多 if(){语句块 1}else if (){语句块 2}…else{语句块 n}语句，在指定条件成立时执行语句块 1，否则再判断第 2 个条件，如果成立执行语句块 2，以此类推，直到所有条件均不成功时执行语句块 n。
- 多分支 switch(){}语句，根据变量或表达式的值与 case 常量匹配情况，选择其中一个分支执行。

### 1. if 语句

if 语句是单分支条件语句，即根据一个条件来控制程序执行的流程。

1）基本语法

```
if (表达式) {
    条件为真时执行代码;
}
```

2）语法说明

if 语句的小括号中表达式的结果类型必须是布尔型，即 true 或 false，当值为 true 时，则执行大括号中的代码；否则跳过大括号中的代码继续执行大括号后面的代码。if 语句的流程图如图 14-11 所示。

【例 14-5-2】判断学生成绩是否及格。代码如下所示，页面效果如图 14-12 所示。

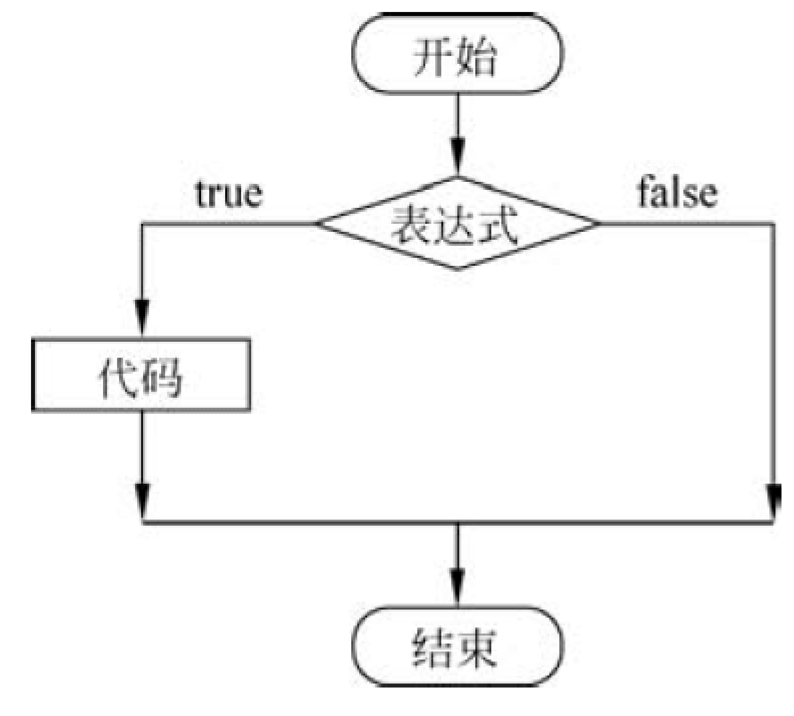



图 14-11　if 条件语句流程图




图 14-12　单 if 语句应用实例



视频讲解

```
1 <!-- edu_14_5_2.html -->
2 <!doctype html>
3 <html lang="en">
4     <head>
5         <meta charset="UTF-8">
6         <title>单if语句的应用</title>
7     </head>
8     <body>
9         <script type="text/javascript">
10            var score = 87;
11            if (score >= 60)
12            {
13                alert("考试成绩为"+score+"分，及格！");
14            }
15        </script>
16    </body>
17 </html>
```

3）代码解释

代码中第 10 行定义变量 score 并赋值为 87；第 11 行判断关系表达式 score>=60，结果为 true，因此执行大括号中的代码，通过告警消息框输出“考试成绩为 87 分，及格！”。

**2．if…else 语句**

If…else 语句是双分支条件语句，即根据一个条件来控制程序执行的流程。

1）基本语法

```
if (表达式) {
    条件成立时执行代码1
}else{
    条件不成立时执行代码2
}
```

2）语法说明

代码中 if…else 语句的小括号中表达式的结果类型必须是布尔型，即 true 或 false，当值为 true 时，则执行代码 1；否则执行代码 2。If…else 语句的流程图如图 14-13 所示。

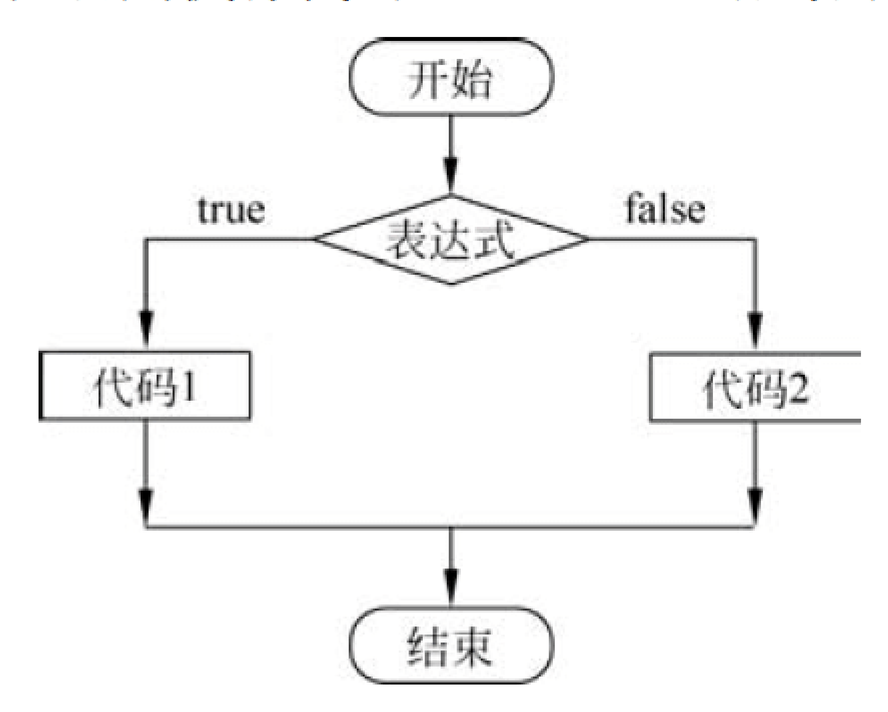



图 14-13　if…else 条件语句流程图

**【例 14-5-3】**判断学生成绩是否及格。代码如下所示，页面效果如图 14-14 所示。

```
<!-- edu_14_5_3.html -->
<!doctype html>
<html lang="en">
    <head>
```



视频讲解

```
5         <meta charset="UTF-8">
6         <title>if...else语句的应用</title>
7     </head>
8     <body>
9         <script type="text/javascript">
10            var score = parseFloat(prompt("请输入课程成绩",50));//解析为实数
11            if (score >= 60)
12            {
13                alert("考试成绩为"+score+"分，及格！");
14            }
15            else
16            {
17                alert("考试成绩为"+score+"分，不及格！");
18            }
19        </script>
20    </body>
21 </html>
```




图 14-14　if…else 语句应用的界面图

代码中第 10 行定义变量 score 并通过提示信息框进行赋值，在输入 50 后，单击“确定”按钮，然后将字符型数据转换浮点数；第 11 行判断关系表达式 score>=60，结果为 false，因此执行第 17 行的代码，通过告警消息框输出“考试成绩为 50 分，不及格！”。如果再次运行程序时，输入 65 后，再单击“确定”按钮，会执行第 13 行语句，通过告警消息框输出“考试成绩为 65 分，及格！”。

**3．多重 if…else 语句**

If…else if…else 语句是多条件多分支语句，可根据两个以上条件来控制程序执行的流程。

1）基本语法

```
if (表达式1) {
    代码1
}
else if (表达式2) {
     代码2
}
...
else {
     代码n
}
```

2）语法说明

在多重 if…else if…else 语句中，if 以及多个 else if 后面的小括号内的表达式的值为 boolean 类型。

程序执行时，按照该语句中表达式的顺序，首先计算表达式 1 的值，如果计算结果为 true，则执行代码 1，执行完后结束 If…else if…else 语句；如果计算结果为 false，则继续

计算表达式 2 的值；以此类推，假设第 n 个表达式值为 true，则执行紧跟的代码 n，并结束 if…else if…else 语句执行；否则继续计算第 n+1 个表达式的值。如果所有表达式的值都为 false，则执行关键字 else 后面的代码 n，结束 if…else if…else 语句的执行。其语句的执行流程如图 14-15 所示。

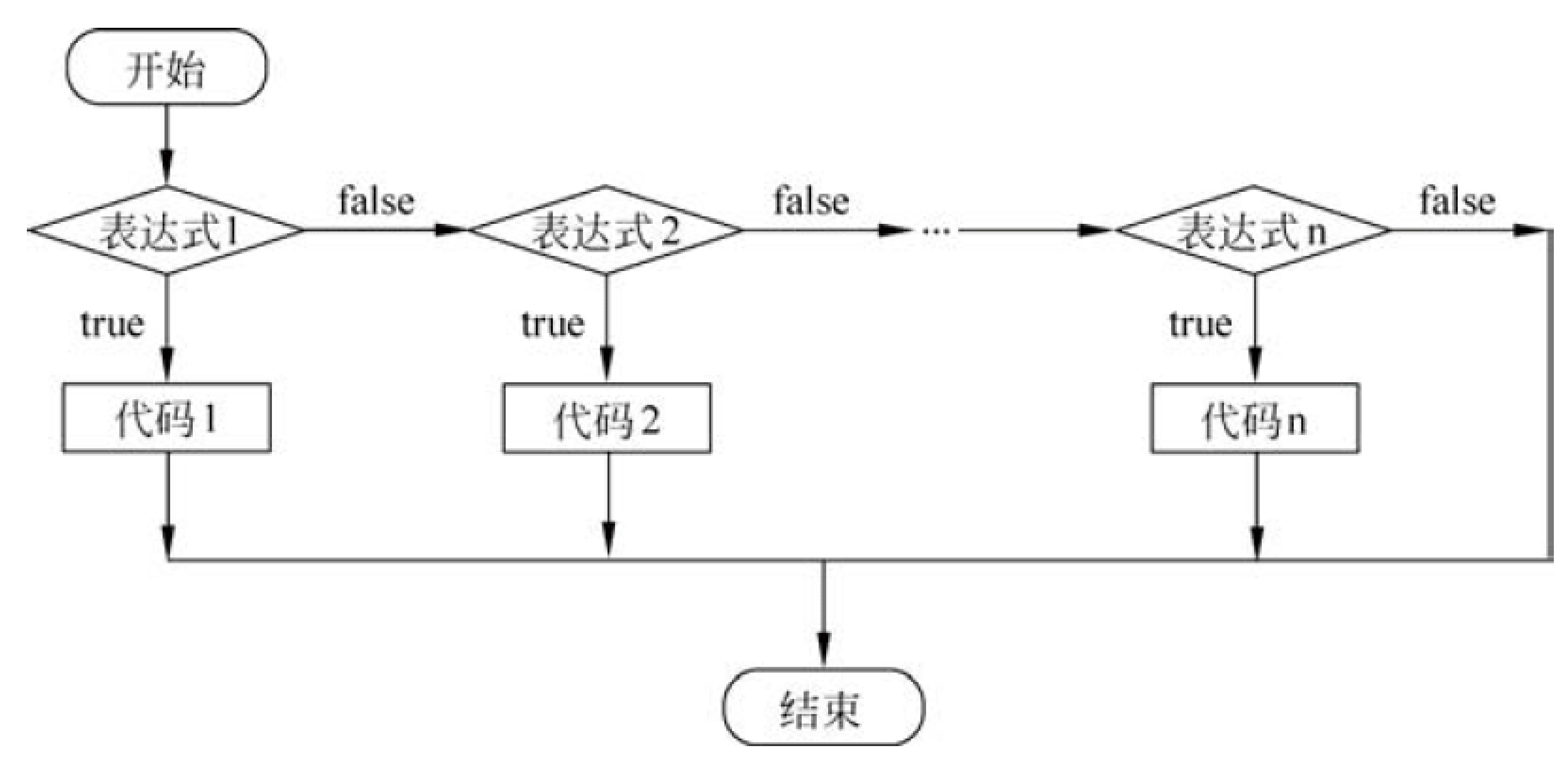



图 14-15　多重 if…else if…else 条件语句流程图

例如，学生成绩五级制表示法中部分 js 代码如下所示：

```
<script type="text/javascript">
    //五级制成绩表示法
    //采用分支嵌套结构
    document.write("<b>判断课程成绩等级</b><br>");
    document.write("课程成绩为85分");
    //var x=85;      //直接给定某课程成绩
    //利用函数输入一个成绩
    var x=prompt("请输入你的成绩：",85);
    if (x!=null){
        if (x>=90) {
            alert("1--成绩为\"优秀\"！");
        }else if (x>=80){
            alert("2--成绩为\"良好\"！");
        }else if (x>=70){
            alert("3--成绩为\"中等\ "！");
        }else if (x>=60){
            alert("4--成绩为\"合格\ "！");
        }else{
            alert("5--成绩为\"不及格\ "！");
        }
    }else{
        alert("请重新输入成绩！");
    }
    alert("6--程序结束！");
</script>
```

**4．switch 语句**

switch 语句是单条件多分支语句，它可以通过判断一个条件完成程序多个分支的控制，比 if…else 使用更为方便，结构更清晰。

**1）基本语法**

```
switch (表达式) {
```

```
    case 常量1:
        {   代码1
          }break;
    case 常量2:
        {    代码2
         }break;
    ...
    case 常量n:
        {   代码n
        }  break;
    default:
        {代码n+1}
}
```

**2）语法说明**

执行 switch 语句时，首先计算变量或表达式的值，然后查找和这个值匹配的 case 常量，如果找到了匹配的常量，则执行后面的语句块；否则执行 default 后的语句块。

在上面的语法格式中，每个case语句块的后面都有一个break语句，其作用是终止switch语句的执行，继续执行 switch 下面的语句。如果没有这个 break 语句，那么 switch 语句会从和表达式的值匹配的 case 常量开始，依次执行后面所有的代码，直到 switch 语句的结尾处。

**【例 14-5-4】**采用 switch 结构实现成绩等级制转百分制。代码如下所示，页面效果如图 14-16 所示。

视频讲解

```
<!-- edu_14_5_4.html -->
<!doctype html>
<html lang="en">
    <head>
        <meta charset="UTF-8">
        <title>switch结构的应用</title>
        <script type="text/javascript">
            function showScore(type){
                var msg="";
                switch (type)
                {
                    case 'A':
                        {msg="成绩为90～100";break;}
                    case 'B':
                        {msg="成绩为80～89";break;}
                    case 'C':
                        {    msg="成绩为70～79";break;}
                    case 'D':
                        {msg="成绩为60～69";break;}
                    case 'E':
                        {msg="成绩低于60";break;}
                    default:
                        {msg="成绩类型错误!";}
                }
                alert(msg);
            }
        </script>
    </head>
    <body>
```

```
30     <form>
31     请输入学生成绩等级：
32     <input type="text" name="score"/><br/>
33     <input type=" button"value=" 显示学生分数"onclick="showScore(score.value)"/>
34     </form>
35     </body>
36 </html>
```

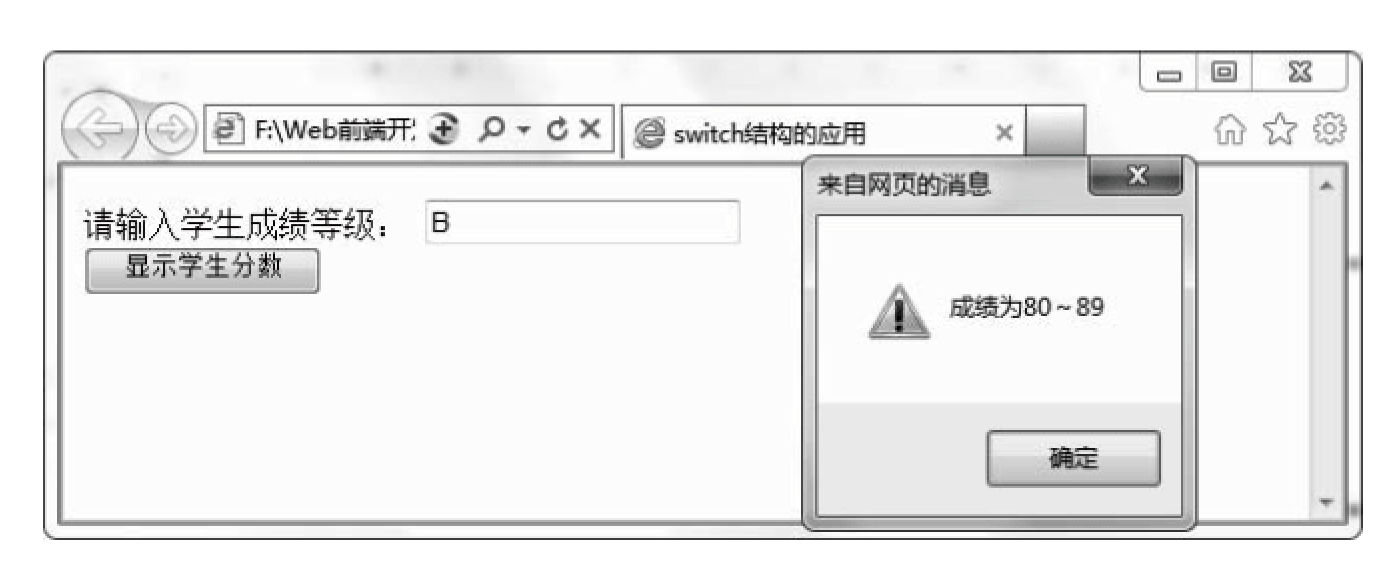



图 14-16　switch 语句的应用

代码中第 7～27 行在 script 标记内定义了 showScore()函数，showScore()函数中使用 switch 结构判断学生成绩等级，并根据 type 的值和 case 后面的常量进行匹配；第 33 行为普通按钮的 onclick 事件属性指定事件处理程序，调用 showScore()函数，并将单行文本输入框 score 中的内容作为函数 showScore()的实际参数。输入等级 B 后，单击“显示学生分数”按钮，弹出告警消息框显示信息“成绩为 80～89”，如图 14-16 所示。

### 14.5.3　循环结构

如果遇到要求将一个班级中所有同学的名字按每行 10 个的方式输出到网页上时，在页面中重复写 n 行相同的代码去输出所有同学的名字，很显然是不科学的，也是不可取的。这种情况使用循环结构可以解决实际问题。JavaScript 提供了 for、while、do…while、for…in 等多种循环。

**1．for 循环**

for 循环是一种结构简单、使用频率高的循环控制语句，作用是有条件地重复执行一段代码。for 语句的执行流程如图 14-17 所示。

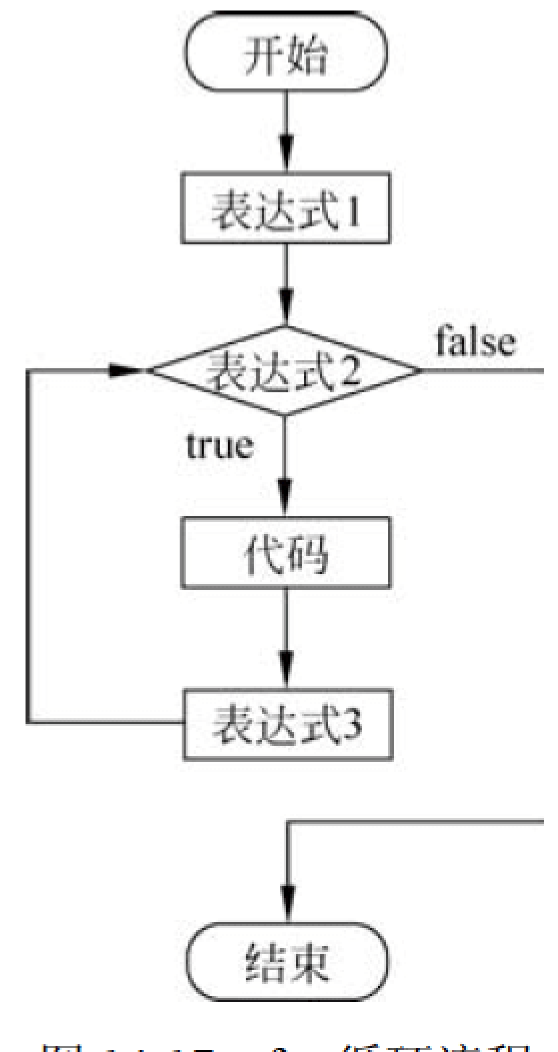



图 14-17　for 循环流程

1）基本语法

```
for (表达式1;表达式2;表达式3)
{
需要循环执行的代码;
}
```

2）语法说明

- for 是 for 语句的关键字，for 关键字后面的一对小括号()不可缺省，括号中用分号“;”分隔的三个表达式。
- 表达式 1 是初始表达式，在循环开始前执行，一般用来定义循环变量。
- 表达式 2 是判断表达式，就是循环条件，必须为 boolean 型数据的表达式，当表达式的结果为 true 时循环继续执行，否则循环结束。
- 表达式 3 是循环表达式，在每次循环执行后都被执行，作用是修改循环变量，然后再进行判断表达式的计算，决定是否继续下一次循环。
- 大括号{}内的代码为循环体，循环体只有一条语句时，大括号{}可以省略。

for 语句的执行规则是：

（1）计算初始表达式的值，完成循环的初始化工作；

（2）计算判断表达式的值，若判断表达式为 true，则转到(3)，否则跳转到(4)；

（3）执行循环体代码，然后计算循环表达式的值，以改变循环变量，跳转到(2)；

（4）结束 for 语句的执行。

**【例 14-5-5】**使用 for 语句计算 1～100 之间所有数字之和。代码如下所示，页面效果如图 14-18 所示。

视频讲解

```
1 <!-- edu_14_5_5.html -->
2 <!doctype html>
3 <html lang="en">
4     <head>
5         <meta charset="UTF-8">
6         <title>for循环的应用</title>
7     </head>
8     <body>
9         <script type="text/javascript">
10            for (var i=1,sum=0; i<=100;i++)
11            {
12                sum = sum + i;
13            }
14            document.write("用for循环求1～100之间所有数字之和为" + sum);
15        </script>
16    </body>
17 </html>
```

3）代码解释

代码中第 10 行设置 for 循环的三个表达式，分别是初始化表达式、判断表达式和循环表达式；第 12 行完成累加功能；第 14 行用 document.write()方法在页面上输出计算结果。




图 14-18　for 循环使用实例

**2. while 循环**

while 循环是 JavaScript 中最基本的循环控制语句之一，其作用是有条件地重复执行某一段代码。

1）基本语法

```
while (表达式)
{
    需要循环执行的代码;
}
```

2）语法说明

while 语句由关键字 while、一对括号()中的表达式和一个大括号{}中的代码组成，代码称为循环体，表达式称为循环条件。由于 while 循环中只有一个判断表达式，不像 for 循环有三个表达式，所以初始化表达式必须挪到 while 循环结构前面，循环表达式必须挪到 while 循环体中。此时 while 循环与 for 循环才能执行同样的功能。

while 语句的执行流程是：

（1）先计算表达式的值，如果值为 true，跳转到（2），否则跳转到（3）；

（2）执行循环体代码，跳转到（1）；

（3）终止 while 语句的执行。

使用 while 循环时需要注意的几个问题是：

- 在 while 循环之前必须完成循环变量初始化工作。
- 不管有没有语句，循环体语句必须使用{}括起来。
- 循环体语句中必须含有循环控制语句，避免发生死循环。

while 语句的执行流程如图 14-19 所示。

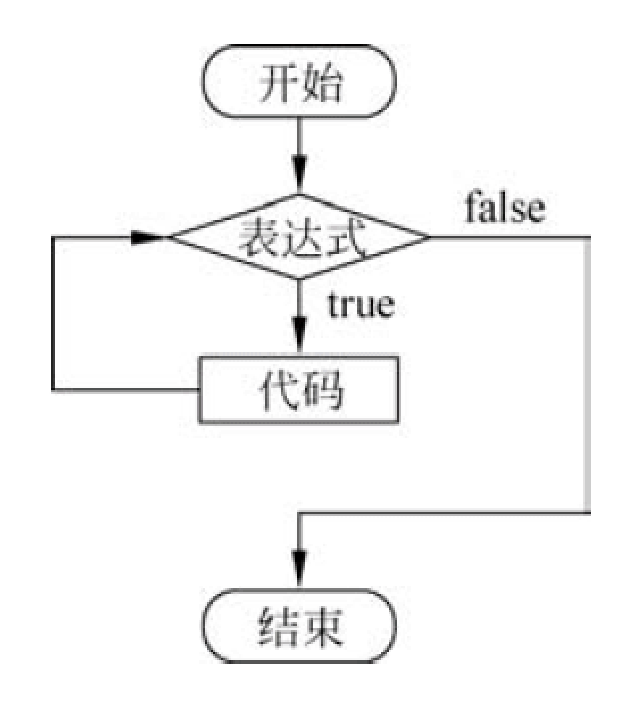



图 14-19　while 循环流程

【例 14-5-6】使用 while 语句计算 1～100 之间所有数字之和。代码如下所示，页面效果如图 14-20 所示。

视频讲解

```
<!-- edu_14_5_6.html -->
<!doctype html>
<html lang="en">
    <head>
        <meta charset="UTF-8">
        <title>while循环的应用</title>
```

```
7      </head>
8      <body>
9          <script type="text/javascript">
10             var i=1,sum=0;       //定义循环初始化表达式
11             while(i<=100)        //定义判断表达式
12             {
13                 sum=sum+i;       //计算累加和
14                 i=i+1;           //定义循环表达式
15             }
16             document.write("用while循环计算1～100之间所有数字之和=" + sum);
17         </script>
18     </body>
19 </html>
```




图 14-20　while 循环使用实例

3）代码解释

代码中第 10 行定义了初始化表达式（相当于 for 循环的第 1 个表达式）；第 11 行设置 while 循环条件为 i<=100（相当于 for 循环的判断表达式），如果条件成立，则将 i 值累加到和 sum 中，第 14 行将 i 的值加 1（相当于 for 循环的循环表达式），直至 i 的值大于 100，跳出循环，并输出 1～100 之间所有数之和。

**3．do…while 循环**

do…while 循环和 while 循环非常类似，也用于重复执行某一段代码。它们的不同点在于：while 循环的条件表达式位于 while 循环的头部，而 do…while 循环的条件表达式位于 do…while 循环的尾部。因此 while 循环总是先检测条件表达式是否成立，如果成立才执行循环体代码；而 do…while 循环则先执行一次循环体内的代码，再判断条件表达式是否成立，如果成立则继续执行循环体语句，否则结束循环。do…while 语句的执行流程如图 14-21 所示。

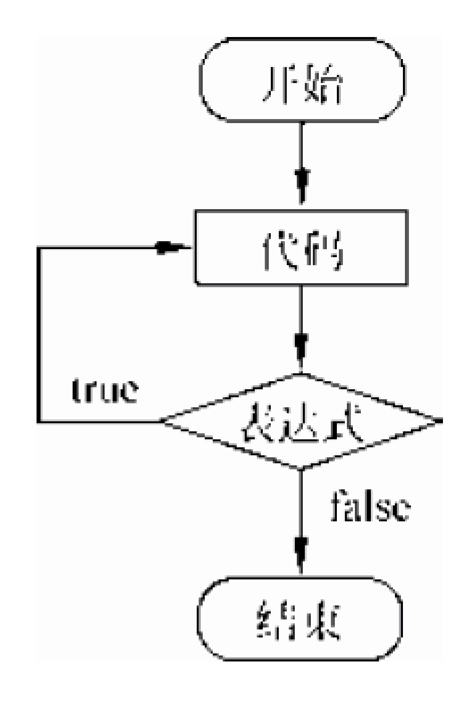



图 14-21　do…while 语句流程

1）基本语法

```
do {
  需要循环执行的代码;
```

```
} while(表达式)
```

2）语法说明

与 while 循环一样，使用 do…while 循环时需要注意的几个问题是：

- 在 do…while 循环之前必须完成循环变量初始化工作。
- 不管有没有语句，循环体语句必须使用{}括起来。
- 循环体语句中必须含有循环控制语句，避免发生死循环。

do…while 循环的执行流程是：

（1）执行循环体代码；

（2）计算表达式的值，如果值为 true，跳转到（1），否则跳转到（3）；

（3）终止 while 语句的执行。

do…while 循环和 while 循环的执行过程基本相同，所不同的是：while 循环先判断给定条件是否成立，后执行循环体；而 do…while 循环则是先执行一次循环体，后判断条件。因此，在一定条件下，while 循环可能一次都不执行，而 do…while 循环在任何条件下至少要执行一次。

**【例 14-5-7】**使用 do…while 循环计算 1～100 的数字之和。代码如下所示，页面效果如图 14-22 所示。

视频讲解

```
<!-- edu_14_5_7.html -->
<!doctype html>
<html lang="en">
    <head>
        <meta charset="UTF-8">
        <title>do-while循环的应用</title>
    </head>
    <body>
        <script type="text/javascript">
            var i=1,sum=0;     //定义初始化表达式
            do                 //先执行一遍循环体语句
            {
                sum= sum + i;  //计算累加和
                i = i + 1;     //定义循环表达式
            }while (i <= 100)  //定义判断表达式
            document.write("用do-while循环计算1～100之间所有数字之和="+sum);
        </script>
    </body>
</html>
```

3）代码解释

【例 14-5-7】和【例 14-5-5】的作用相同，都是求 1～100 的数字之和。




图 14-22　do…while 语句使用实例

**4．for…in 循环**

在 JavaScript 中，除了 for 语句可以用于控制循环结构以外，还有另一种形式的 for 语

句，主要用于数组、集合对象的遍历，也需要循环执行某一段代码，即 for…in 循环。

1）基本语法

```
for (variable in object)
{
    需要循环执行的代码;
}
```

2）语法说明

variable 可以是一个变量名、数组元素或对象属性，object 可以是一个对象名或者计算结果为对象的表达式。for…in 循环将对 object 对象的每一个属性或每一元素都执行一次循环，在循环过程中，首先将 object 对象的一个属性名作为字符串赋给变量 variable，这样在循环体内就可以通过变量 variable 访问对象属性。

**【例 14-5-8】** 使用 for…in 循环列出 Screen 对象的所有属性和数组对象的所有元素。代码如下所示，页面效果如图 14-23 所示。

视频讲解

```
01 <!-- edu_14_5_8.html -->
02 <!doctype html>
03 <html lang="en">
04     <head>
05         <meta charset="UTF-8">
06         <title>for-in循环的应用</title>
07     </head>
08     <body>
09         <script type="text/javascript">
10             var i = 1;  //定义计数器变量
11             document.write("<h3>document对象所有属性名称/属性值：</h3>");
12             //遍历对象的所有属性
13             for (var property in screen)
14             {
15                 document.write(i+"."+property+"/"+screen[property]+"  ");
16                 if (i % 2 ==0) {document.write("<br/>");} //每行输出两对
17                 i++;
18             }
19             //遍历数组对象的所有元素
20             var stu=new Array("王春平","张宏伟","金一鑫","李大为","任小月","储忠庆");
21             var j=1;//定义计数器j
22             document.write("<h3>数组的元素分别为：</h3>");
23             for (var student in stu)
24             {
25                 document.write(j+"."+stu[student]+"  ");
26                 if (j % 2 ==0) {document.write("<br/>");} //每行输出两对
27                 j++;
28             }
29         </script>
30     </body>
31 </html>
```

3）代码解释

代码中第 12 行～第 18 行使用变量 property 遍历输出屏幕 screen 对象的所有属性及属性值。第 19 行～第 28 行使用变量 student 遍历输出 stu 数组对象的所有元素。

**5．循环的嵌套**

一个循环内又包含着另一个完整的循环结构，称为循环嵌套。内嵌的循环中还可继续

嵌套别的循环，这就构成多重循环结构。




图 14-23　for…in 循环使用实例

【例 14-5-9】计算 1！+2！+3！+4！+5！的和。代码如下所示，页面效果如图 14-24 所示。

视频讲解

```
1 <!-- edu_14_5_9.html -->
2 <!doctype html>
3 <html lang="en">
4     <head>
5         <meta charset="UTF-8">
6         <title>计算1！+2！+3！+4！+5！的和</title>
7     </head>
8     <body>
9         <script type="text/javascript">
10        document.write("计算1！+2！+3！+4！+5！的和<br/>");
11        for (i=1,sum=0;i<=5 ;i++ )
12        {
13            for (j=1,cj=1;j<=i ;j++ )
14            {
15                cj=cj*j;
16            }
17            document.write(i+"!="+cj+"<br/>");
18            sum=sum+cj;
19        }
20        document.write("1！+2！+3！+4！+5！="+sum);
21        </script>
22    </body>
23 </html>
```

**代码解释**

代码中第 11～19 行是外层 for 循环，计算连续若干个数的阶乘的和；第 13～16 行是内层 for 循环，计算 N!。第 17 行外循环每执行 1 次就输出循环变量的阶乘值。




图 14-24　计算 1！+2！+3！+4！+5！的和

### 6．循环中断与继续

在正常的循环结构中，每次循环都是从满足条件开始直到不满足条件结束，也就是说，必须完整地执行完所有的循环。但在实际问题中有时并不需要完整执行完所有循环才结束程序，可能遇到一些需要提前中止或跳过某些循环等情况，这时需要使用 break 和 continue 语句来解决实际问题。

在循环体中 break 语句的作用是立即结束循环，跳转到循环后面的语句，而不管原来的循环还有多少次，都不会再执行。在循环体中 continue 语句的作用是结束本次循环，本次循环后面的所有语句都不会执行，直接进入下一次循环，直到循环结束。

**【例 14-5-10】**计算 5！+6！+…+n！的和(5⩽n⩽15)。代码如下所示，页面效果如图 14-25 和图 14-26 所示。

视频讲解

```
<!-- edu_14_5_10.html -->
<!doctype html>
<html lang="en">
    <head>
        <meta charset="UTF-8">
        <title>计算部分∑N!的和</title>
    </head>
    <body>
        <script type="text/javascript">
            document.write("计算部分∑N!的和<br/>");
            var n = prompt("请输入整数N: ",20);
            for (i=1,sum=0;i<=n ;i++ )
            {
                if (i>15) {break;}  //当循环到第15次时跳出循环
                //当i为1-5之间的数时结束本次循环进入下1次循环
                if (i>=1 && i<5){
                    continue;
                }else{    //当i大于等于5时执行循环
                    for (j=1,cj=1;j<=i ;j++ )
                    {
                        cj=cj*j;      //计算阶乘
                    }
                    document.write(i+"!="+cj+"<br/>");
                    sum=sum+cj;    //累加阶乘之和
                }
            }
            i=i-1
            document.write("∑"+i+"!="+sum);
        </script>
    </body>
</html>
```

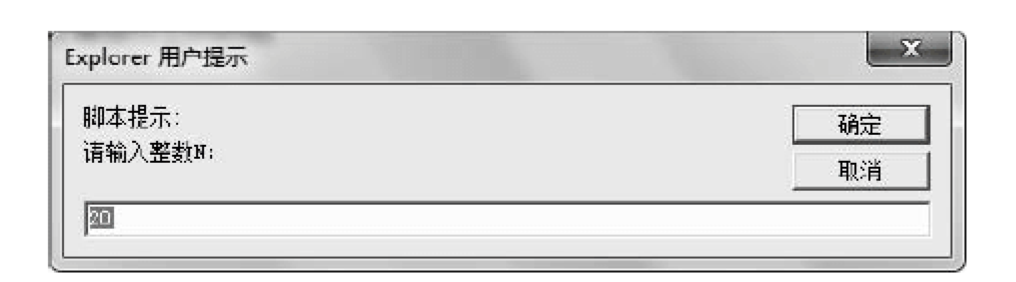



图 14-25　提示信息框界面

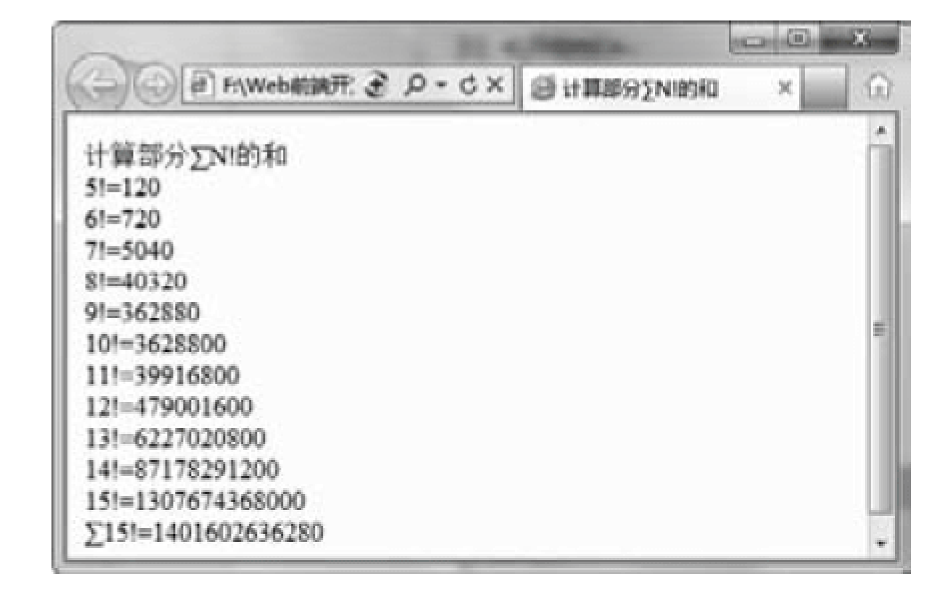



图 14-26　计算 5！+6！+…+N！的和

运行代码后，首先弹出提示信息框，要求输入整数 N 的值，默认值为 20，如图 14-25 所示。单击“确定”按钮后，N 取默认值 20 后，计算部分∑N!的和，如图 14-26 所示。

**代码解释**

代码中第 11 行定义变量 n，并通过 prompt()方法给变量 n 赋值（设默认值为 20）；第 12～26 行是外层 for 循环，计算连续若干个数的阶乘的和；第 14 行采用单分支 if 语句判断当变量 i 的值大于 15 时执行 break 语句，立即结束循环，即跳出外层循环，从第 27 行代码开始执行，如果 i 的值小于等于 15 时，则继续执行外循环；第 16～25 行采用双分支 if…else 结构根据变量 i 的取值范围是否在[1，4]之间来判断是否结束本次循环直接进入下一次循环。如果在此区间内，执行 continue 语句，结束本次循环，后面所有语句此次不再执行，开始下一次循环，直到结束；如果不在此区间，则执行内循环；第 19～22 行是内层 for 循环，计算变量 j!，结果保存在变量 cj 里。第 23 行外循环每执行一次就输出循环变量的阶乘值。

# 14.6 JavaScript 函数

JavaScript 函数分为系统内部函数和系统对象定义的函数及用户自定义函数。函数就是完成一个特定的功能的程序代码。函数只需要定义一次，可以多次使用，从而提高程序代码的复用率，既减轻开发人员的负担，又降低了代码的重复度。

函数需要先定义后使用，JavaScript 函数一般定义在 HTML 文件的头部 head 标记或外部 JS 文件中，而函数的调用可以在 HTML 文件的主体 body 标记中的任何位置。

## 14.6.1 常用系统函数

JavaScript 中有许多预先定义的系统内部函数和对象定义的函数，如 document.write()就是其中之一。这些预定义的系统函数大多数存在于预定义的对象中，例如 String、Date、Math、window 及 document 对象中都有很多预定义的函数，只有熟练使用这些函数才能充分发挥 JavaScript 的强大功能，简洁、高效地完成程序设计任务。

常用系统函数分全局函数和对象定义的函数。全局函数它不属于任何一个内置对象，使用时不需要加任何对象名称，直接调用。例如 eval()、escape()、unescape()、parseFloat()、parseInt()、isNaN()等。全局函数如表 14-13 所示。对象定义的函数依赖于对象，使用时需要加对象名称（顶层对象 window 除外），例如 alert()、confirm()、prompt()等函数是 window 对象定义的函数。document.write()是 document 对象的方法。

**表 14-13　全局函数名称与说明对照表**

| 函　数 | 说　明 |
| --- | --- |
| decodeURI() | 解码某个编码的 URI |
| decodeURIComponent() | 解码一个编码的 URI 组件 |
| encodeURI() | 把字符串编码为 URI |
| encodeURIComponent() | 把字符串编码为 URI 组件 |
| eval() | 计算 JavaScript 字符串，并把它作为脚本代码来执行 |
| escape() | 对字符串进行编码 |
| unescape() | 对由 escape()编码的字符串进行解码 |

续表

| 函　　数 | 说　　明 |
|---|---|
| parseFloat() | 解析一个字符串并返回一个浮点数 |
| parseInt() | 解析一个字符串并返回一个整数 |
| getClass() | 返回一个 JavaObject 的 JavaClass |
| isNaN() | 检查某个值是否是非数 |
| isFinite() | 检查某个值是否为有穷大的数 |
| Number() | 把对象的值转换为数字 |
| String() | 把对象的值转换为字符串 |

下面重点介绍常用的全局函数和常用的对象函数。

**1．全局函数**

1）计算表达式的结果函数

（1）基本语法。

```
eval (string)
```

（2）语法说明。

eval()函数的作用是返回字符串 string 表达式中的值。该函数接受原始字符串作为参数，将该字符串作为代码在上下文环境中执行，并返回执行结果。

（3）参数说明。

string：要计算的字符串表达式，含有要计算的 JavaScript 表达式或要执行的语句。

**【例 14-6-1】**eval 函数的应用。代码如下所示，页面效果如图 14-27 所示。

视频讲解

```
1 <!-- edu_14_6_1.html -->
2 <!doctype html>
3 <html lang="en">
4     <head>
5         <meta charset="UTF-8">
6         <title>eval()函数的应用</title>
7     </head>
8     <body>
9         <h4>eval()函数的应用</h4>
10        <script type="text/javascript">
11            eval("x=20;y=30;document.write('x='+x+',y='+y+',x*y='+x*y)");
12            document.write("<br/>");
13            document.write("2+2="+eval("2+2"));
14            var abce;  //声明变量未赋值
15            document.write("<br/>abce="+eval(abce));
16        </script>
17    </body>
18 </html>
```

（4）代码解释。

代码中第 11 行中 eval()函数的参数是代码，在上下文环境下执行，并返回 x*y 的计算结果，并输出到页面上；第 13 行中 eval()函数的参数是表达式，计算 2+2 的值并输出到页面上；第 15 行中 eval()函数的参数不是表达式，而是只定义未赋值的变量 abce，所以返回结果为 undefined。




图 14-27　eval()函数使用实例

2）编码与解码函数

- 编码函数 escape()。

（1）基本语法。

```
escape (string)
```

（2）语法说明。

escape()函数可对字符串（ISO-Latin-1 字符集）进行编码，这样就可以在所有的计算机上读取该字符串。该函数不会对 ASCII 字符和数字进行编码，也不会对下面这些 ASCII 标点符号进行编码：* @ – _ + . / 。其他所有的字符都会被转义序列替换。

（3）参数说明。

string：要被转义或编码的字符串。

**【例 14-6-2】**escape()函数的应用。代码如下所示，页面效果如图 14-28 所示。





视频讲解

图 14-28　escape()函数使用实例

```
<!-- edu_14_6_2.html -->
<!doctype html>
<html lang="en">
    <head>
        <meta charset="UTF-8">
        <title>escape()函数的应用</title>
    </head>
    <body>
        <h4>escape()函数的应用</h4>
        <script type="text/javascript">
            document.write("\"?\"进行编码后为:"+escape("?")+"<br/>");
            document.write("\"JavaScript编程!\"编码后为: "+escape("JavaScript
             编程!"));
            document.write("<br/>Mr Chu 你好!"+"编码后为:"+escape("Mr Chu
            你好!"));
        </script>
    </body>
</html>
```

（4）代码解释。

代码中第 11 行中对字符“?”进行编码，由于“?”的 ASCII 码为 63，转换为十六进制为 3F，因此输出“%3F”；第 12 行对字符串“JavaScript 编程!”进行编码，这当中“JavaScript”是字母，因此不编码，而其他字符则进行了编码，汉字是双字节编码，格式为“%u”+两个字节的十六进制数据，如“编”的编码%u7F16；第 13 行对空格和汉字进行编码，空格编码为%20。

- 解码函数 unescape()。

（1）基本语法。

```
unescape (string)
```

（2）语法说明。

unescape()函数返回的字符串是 ISO-Latin-1 字符集的字符。该函数通过找到形式为%xx 和%uxxxx 的字符序列（x 表示十六进制的数字），用字符 \u00xx 和 \uxxxx 替换这样的字符序列进行解码。

（3）参数说明。

string：要解码或反转义的字符串。

**【例 14-6-3】**使用 unescape()函数对经过编码的字符进行解码。代码如下所示，页面效果如图 14-29 所示。

视频讲解




图 14-29　unescape()函数使用实例

```
1 <!-- edu_14_6_3.html -->
2 <!doctype html>
3 <html lang="en">
4     <head>
5         <meta charset="UTF-8">
6         <title>unescape()函数的应用</title>
7     </head>
8     <body>
9         <h4>unescape()函数的应用</h4>
10        <script type="text/javascript">
11            document.write("\"%3F\"解码后为: " + unescape("%3F") + "<br/>");
12            document.write("JavaScript%u7F16%u7A0B%21解码后为: "+unescape
              ("JavaScript%u7F16%u7A0B%21"));
13        </script>
14    </body>
15 </html>
```

（4）代码解释。

代码中第 11 行中对“%3F”进行解码，得到的结果为“？”；第 12 行对字符串“JavaScript%u7F16%u7A0B%21”进行解码，得到的结果为“JavaScript 编程！”。

3）字符型转换成数值型函数。

- parseFloat()函数。

（1）基本语法。

```
parseFloat (string)
```

（2）语法说明。

parseFloat()函数的作用是返回 string 字符串对应的实数值。只有字符串中的第一个数字会被返回，如果字符串 string 的第一个字符不能被转换为数字，那么 parseFloat()函数会返回 NaN。

（3）参数说明。

string：要被解析的字符串。

【例 14-6-4】parseFloat()函数的应用。代码如下所示，其页面效果如图 14-30 所示。

视频讲解

```
1 <!-- edu_14_6_4.html -->
2 <!doctype html>
3 <html lang="en">
4     <head>
5         <meta charset="UTF-8">
6         <title>parseFloat()函数的应用</title>
7     </head>
8     <body>
9        <h4>parseFloat()函数的应用</h4>
10       <script type="text/javascript">
11         document.write("\"100\"转换后为: "+parseFloat("100")+"<br/>");
12         document.write("\"100.00\"转换后为: "+parseFloat("100.00")+"<br/>");
13         document.write("\"100.88\"转换后为: "+parseFloat("100.88")+"<br/>");
14         document.write("\"12 34 56\"转换后为: "+parseFloat("12 34 56")+"<br/>");
15         document.write("\" 60 \"转换后为: "+parseFloat(" 60 ")+"<br/>");
16         document.write("\"40 years\"转换后为:"+parseFloat("40 years")+"<br/>");
17         document.write("\"衣服100元\"转换后为:"+parseFloat("衣服100元")+"<br/>");
18       </script>
19    </body>
20 </html>
```

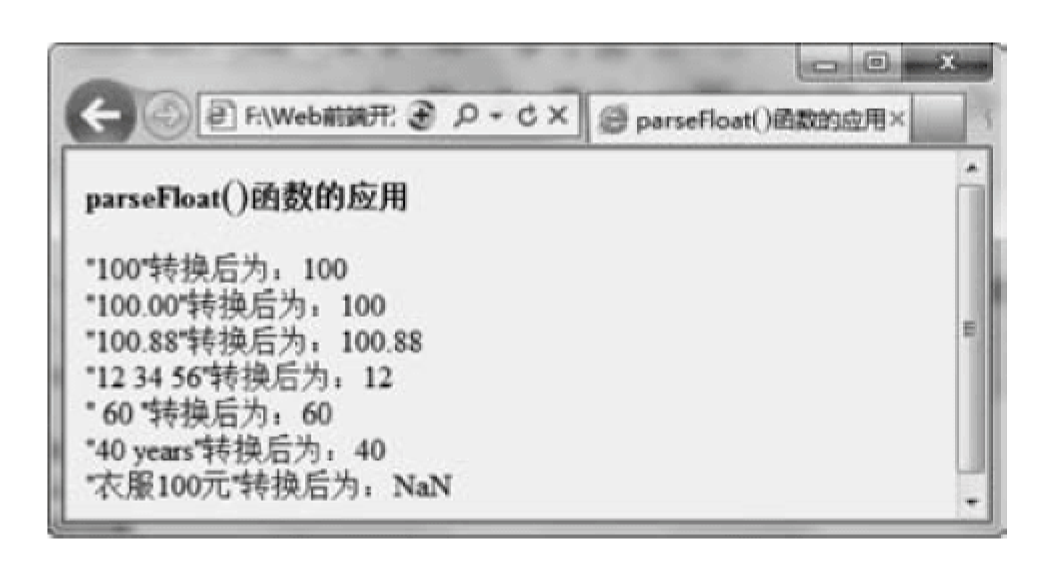



图 14-30　parseFloat()函数使用实例

（4）代码解释。

代码中第 11 行将一个整数 100 字符串转换为实数，输出 100；第 12 行转换一个实数 100.00 字符串转化为实数，由于小数部分为 0.00，因此输出时被省略，结果为 100；第 13 行的实数 100.88 字符串转换为 100.88，输出也是 100.88；第 14 行有三个用空格分隔的整数字符串，解析后只能将第 1 个数 12 转换为实数，空格开始向后全部忽略，所以输出为 12；第 15 行的数字 60 前后各有一个空格，转换时前面的空格被忽略，因此输出 60；第 16 行只转换空格前的数字，输出 40；第 17 行第 1 个是字符不能被转换为数字，因此输出 NaN。

- parseInt()函数。

（1）基本语法。

```
parseInt (string,radix)
```

（2）语法说明。

parseInt()函数的作用是返回 string 字符串对应的十进制整数值，参数 radix 用于指定数字的基数。只有字符串中的第 1 个数字会被返回，如果字符串的第 1 个字符不能被转换为数字，那么 parseInt()函数会返回 NaN。

（3）参数说明。

parseInt()函数的参数及说明如表 14-14 所示。

**表 14-14　parseInt()函数参数及说明**

| 参　数 | 说　明 |
|---|---|
| string | 要被解析的字符串 |
| radix | 表示要解析的数字的基数。该值介于 2～36 之间<br>如果省略该参数或其值为 0，则数字将以 10 为基数来解析<br>如果它以 0 开头，将以 8 为基数<br>如果它以 0x 或 0X 开头，将以 16 为基数<br>如果该参数小于 2 或者大于 36，则 parseInt()将返回 NaN |

**【例 14-6-5】**parseInt()函数的应用。代码如下所示，页面效果如图 14-31 所示。

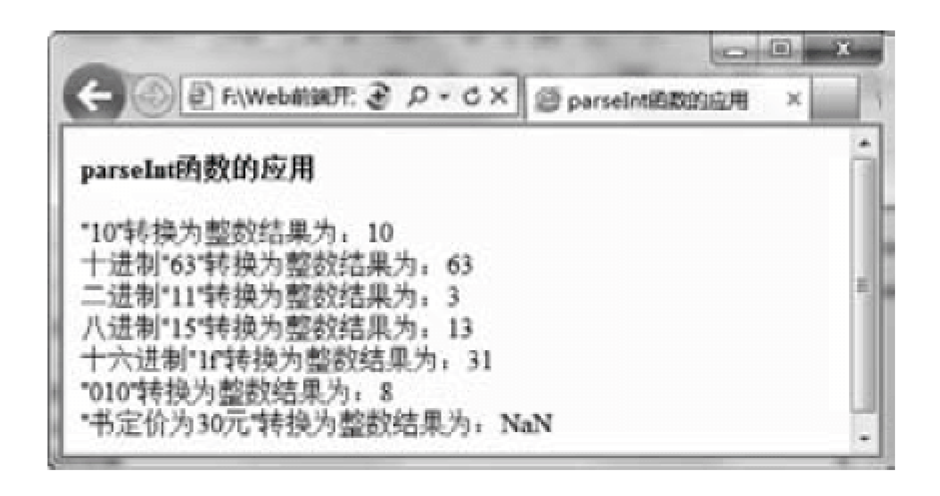



图 14-31　parseInt()函数使用实例

```
<!-- edu_14_6_5.html -->
<!doctype html>
<html lang="en">
 <head>
     <meta charset="UTF-8">
     <title>parseInt()函数的应用</title>
 </head>
 <body>
     <h4>parseInt()函数的应用</h4>
     <script type="text/javascript">
       document.write("\"10\"转换为整数结果为: "+parseInt("10")+"<br>");
       document.write("十进制\"63\"转换为整数结果为:"+parseInt("63",10)+"<br>");
       document.write("二进制\"11\"转换为整数结果为:"+parseInt("11",2)+
      "<br>");
       document.write("八进制\"15\"转换为整数结果为:"+parseInt("15",8)+
       "<br>");
       document.write("十六进制\"1f\"转换为整数结果为: "+parseInt("1f",16)+
       "<br >");
```

```
16          document.write("\"010\"转换为整数结果为:"+parseInt("010")+"<br>");
17          document.write("\"书定价为30元\"转换为整数结果为:"+parseInt("书定价
            为30元")+"<br />");
18          </script>
19      </body>
20  </html>
```

（4）代码解释。

代码中第 11 行省略函数的第 2 个参数，默认为十进制，输出结果为 10；第 12 行指定 63 为十进制，输出结果为 63；第 13 行指定 11 为二进制，输出结果为 3；第 14 行指定 15 为八进制，输出结果为 13；第 15 行指定 1f 为十六进制，输出结果为 31；第 16 行 010 以 0 开头，表示是八进制，输出结果为 8；第 17 行第 1 个字符不能被转换为数字，输出 NaN。

4）判断是否是 NaN 函数

（1）基本语法。

```
isNaN (string)
```

（2）语法说明。

isNaN()函数的作用是判断 string 是否为数值，如果 string 是特殊的非数字值 NaN（或者能被转换为这样的值），返回的值就是 true。如果 string 是其他值，则返回 false。

（3）参数说明。

string：要检测的值。

**【例 14-6-6】**isNaN()函数的应用。代码如下所示，页面效果如图 14-32 所示。

```
1  <!-- edu_14_6_6.html -->
2  <!doctype html>
3  <html lang="en">
4      <head>
5          <meta charset="UTF-8">
6          <title>isNaN()函数的应用</title>
7      </head>
8      <body>
9          <h4>isNaN()函数的应用</h4>
10         <script type="text/javascript">
11             document.write("\"40\"是否是非数值:"+isNaN(40)+"<br>");
12             document.write("\"3*30\"是否是非数值:"+isNaN(3*30)+"<br>");
13             document.write("\"JavaScript\"是否是非数值:"+isNaN("JavaScript"));
14         </script>
15     </body>
16 </html>
```

视频讲解

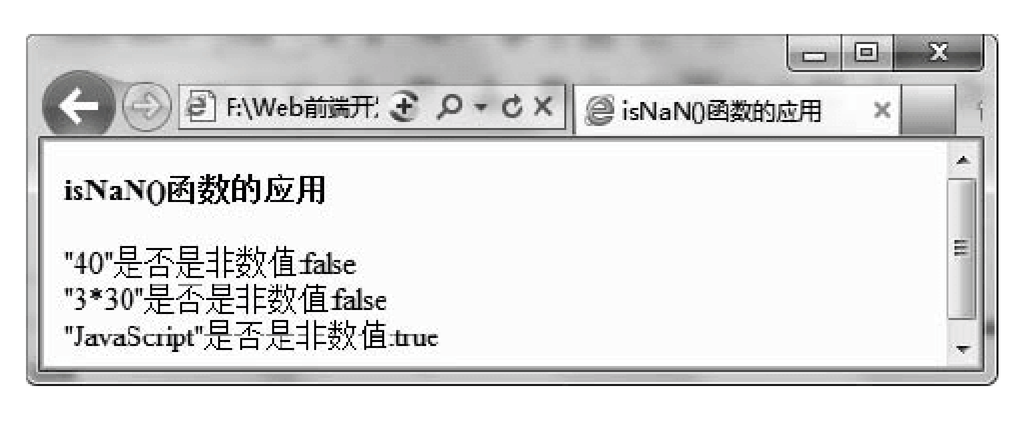



图 14-32　isNaN()函数使用实例

（4）代码解释。

代码中第 11 行的 40 是数值，返回 false；第 12 行的“3*30”可以转换为数值，返回

false；第 13 行的 JavaScript 不可以转换为数值，返回 true。

**2．常用的对象函数**

（1）toString(radix)。将 Number 型数据转换为字符型数据，并返回指定的基数的结果。其中 radix 范围 2～36，若省略该参数，则使用基数 10。例如：

```
var a = 12;
alert(a.toString(2));    //告警框输出结果为1100（二进制）
alert(a.toString());     //告警框输出结果为12(默认的十进制)
```

（2）toFixed(n)。将浮点数转换为固定小数点位数的数字。n 是整数，设置小数的位数，如果省略了该参数，将用 0 代替。例如：

```
var a = 2016.1567;
alert(a.toFixed(2));     //保留2位小数，告警框输出结果为2016.16
alert(a.toFixed(5));     //保留5位小数，告警框输出结果为2016.15670
```

（3）字符串查找和提取常用函数。

字符串对象提供了一系列字符串查找和提取的方法，如表 14-15 所示。

**表 14-15　String 对象查找与提取方法及说明**

| 方　法 | 说　明 |
|---|---|
| indexOf(searchvalue,fromindex) | 从前向后搜索字符串。返回某个指定的字符串值在字符串中首次出现的位置，如果没有发现，返回-1 |
| lastIndexOf(searchvalue,fromindex) | 从后向前搜索字符串。返回一个指定的字符串值最后出现的位置，如果没有发现，返回-1 |
| charAt(index) | 返回在指定位置的字符 |
| substring(start,stop) | 用于提取从 start 到 end（不包括该元素）之间的字符串 |

在开发 Web 应用程序过程中，经常需要对用户输入的数据进行有效性、合法性验证。通过程序提取用户输入的数据，然后对提取的字符串进行适当处理。如判断用户名首字符是否为字母、字符串中是否包含特定字符等，通过 String 对象提供的方法可以很容易实现。举例如下：

```
var str="Welcome to you!";
var substr=str.substring(3,6); //从第0个字符开始数，第3个到第6个之间字符为"com"
var somestr=str.charAt(4);     //从第0个字符开始数，取第4个字符结果是"o"
```

【例 14-6-7】判断邮箱地址的合法性。代码如下所示，页面效果如图 14-33 所示。

视频讲解

```
<!-- edu_14_6_7.html -->
<!doctype html>
<html lang="en">
   <head>
       <meta charset="UTF-8">
       <title>验证邮箱的合法性</title>
       <script type="text/javascript">
           function emailCheck()
           {
               //获取用户输入的邮箱地址相关信息,仅验证最多有两个"."号
               var emailString=document.form1.email.value;
               var emailLength=emailString.length;
```

```
13                  var index1=emailString.indexOf("@");
14                  var index2=emailString.indexOf("."); //取第1个点
15                  var index3=emailString.lastIndexOf(".");//取最后一个点
16                  var msg="验证邮箱地址实例:\n\n";
17                  msg+="邮箱地址:"+emailString+"\n";
18                  msg+="验证信息:";
19                  var emailFlag=false;
20                  /*    返回相关验证信息
21                  首先判断邮箱中必须有@和.，且@必须在前，.在后;
22                  @不能为首字符，.不能为最后一个字符*/
23                  if (index1!=-1 && index3!=-1 && index3>index1+2 && index1!=0
                    && index3!=emailLength-1)
24                  { //然后判断有几个点号
25                      if (index2==index3)
26                      { //只有一个
27                         emailFlag= (index2>=index1+3)?true:false;
28                      }else{//有两个以上
29                         emailFlag=(index3>=index2+3 && index2>=index1+3)?
                           true:false;
30                      }
31                  }else{
32                           emailFlag=false;
33                  }
34                      if(!emailFlag)
35                      {
36                      msg+="邮箱地址不合法!\n\n"
37                      msg+="不能同时满足如下条件:\n";
38                      msg+="1、邮件地址中同时含有'@'和'.'字符; \n";
39                      msg+="2、'@'后必须有'.'，且中间至少间隔一个字符; \n"
40                      msg+="3、'@'不为第一个字符，'.'不为最后一个字符。\n"
41                      }
42                      else
43                      {
44                      msg+="邮箱地址合法!\n\n";
45                      msg+="能同时满足如下条件:\n";
46                      msg+="1、邮件地址中同时含有'@'和'.'字符; \n";
47                      msg+="2、'@'后必须有'.'，且中间至少间隔一个字符; \n"
48                      msg+="3、'@'不为第一个字符，'.'不为最后一个字符。\n"
49                  }
50                  alert(msg);
51              }
52          </script>
53      </head>
54      <body>
55          <form name="form1">
56              邮箱地址:<input type="text" name="email" value="@">
57              <input type="button" value="验证邮箱地址" onclick="emailCheck()">
58          </form>
59      </body>
60  </html>
```

上述代码中第 8～51 行定义了一个 JavaScript 函数名为 emailCheck()；第 56 行定义了一个文本输入框给用户输入 Email 地址；第 57 行定义了一个普通按钮并为按钮设置了 onClick 事件句柄。在文本输入框中输入 Email，单击按钮时会触发 Click 事件调用 emailCheck()函数验证 Email 是否符合标准。如果输入的 Email 是不合法的，如 asd@1223e，

则弹出“邮箱地址不合法！”的信息；如果输入的 Email 是合法的，如 testemial@163.com，则弹出“邮箱地址合法”的信息。




图 14-33　验证邮箱合法性时告警信息窗口

## 14.6.2　自定义函数

### 函数定义

函数是由事件驱动的或者当它被调用时执行的可重复使用的代码块。

1）基本语法

```
function functionname(argument1,argument2,..., argumentn)
{
    这里是要执行的代码（也称为函数体）;
}
```

2）语法说明

- 函数就是包括在大括号中的代码块，使用关键词 function 来定义。当调用该函数时，会执行函数内的代码。
- 在调用函数时，可以向其传递值，这些值被称为参数。这些参数可以在函数中使用。可以发送任意多的参数，参数之间由逗号分隔。也可以没有参数，但括号不能省略，参数类型不需要给定。
- 函数体必须写在“{”和“}”内，“{”、“}”定义了函数的开始和结束。
- JavaScript 中区分字母大小写，因此“function”这个词必须全部字母小写，否则程序就会出错。另外需要注意的是，必须使用大小写完全相同的函数名来调用函数。

**【例 14-6-8】**自定义求梯形面积的函数。代码如下所示，页面效果如图 14-34 所示。





视频讲解



图 14-34　自定义函数使用实例

```
1 <!-- edu_14_6_8.html -->
2 <!doctype html>
3 <html lang="en">
4     <head>
5         <meta charset="UTF-8">
6         <title>自定义函数的应用</title>
7         <script type="text/javascript">
8             function area(a,b,c){
9                 s=((parseInt(a.value)+parseInt(b.value))/2*parselnt(c.value);
10                alert("梯形的面积为"+s);
11            }
12        </script>
13    </head>
14    <body>
15        <form>
16            上底:<input type="text" name="a"><br/>
17            下底:<input type="text" name="b"><br/>
18            高度:<input type="text" name="c"><br/>
19                 <input type="button" onclick="area(a,b,c)" value="求面积"><br/>
20        </form>
21    </body>
22 </html>
```

3）代码解释

代码中第 8～11 行定义了有参数的 JavaScript 函数 area（a,b,c），三个参数分别表示梯形的上底、下底和高；第 9 行将三个文本框中的输入数据转换成整数；第 10 行通过告警消息框输出面积；第 15～20 行在 body 标记中插入表单，在表单中插入表单元素：三个文本输入框和一个普通按钮，文本输入框用于输入梯形的上底、下底和高；普通按钮用于调用自定义函数 area(a,b,c)，第 19 行给普通按钮的 onclick 事件绑定事件处理程序，完成梯形面积的计算。

## 14.6.3 带参数返回的 return 语句

如果需要返回函数的计算结果，可以使用带参数的 return 语句。如果不需要返回函数的计算结果，则使用不带参数的 return 语句。

### 1. 基本语法

```
return 函数执行结果;   //有返回值
return ;              //无返回值,此句可有可无
```

### 2. 语法说明

- 有值返回的函数调用方式与无值返回的调用方式略有不同。无值返回可以通过事件触发、程序触发等方式调用；有值返回的函数类似于操作数，和表达式一样时可以直接参加运算，不需要通过事件或程序来触发。
- 函数体内使用不带返回值的 return 语句可以结束程序运行，其后所有语句均不再执行。return 语句只能返回一个计算结果。return 语句后可以跟上一个具体的值，也可以是一个变量，还可以是一个复杂的表达式。

【**例 14-6-9**】return 语句的应用。代码如下所示，页面效果如图 14-35 所示。

视频讲解

```
 1 <!-- edu_14_6_9.html -->
 2 <!doctype html>
 3 <html lang="en">
 4     <head>
 5         <meta charset="UTF-8">
 6         <title>return语句返回计算结果</title>
 7         <script type="text/javascript">
 8         function showSum(){
 9             document.write("3+4+5结果为："+plus(3,4,5));  //调用函数
10             return;              //结束函数
11         }
12         function plus(a,b,c){
13             return a+b+c;        //返回函数结果
14         }
15         </script>
16     </head>
17     <body>
18         <h4>计算3个数的和</h4>
19         <script type="text/javascript">
20             showSum();
21         </script>
22     </body>
23 </html>
```

**3．代码解释**

代码中第 8～14 行定义了两个函数，分别是一个带有三个形式参数的函数 plus(a,b,c)和不带参数的函数 showSum()；在 plus(a,b,c)函数体中就一条 return 语句，返回三个数累加和；在 showSum()函数体中先调用 plus(3,4,5)函数，然后用 return 结束函数的运行；第 20 行执行 showSum()函数。在程序中两种 return 语句均使用了，但作用却不相同。




图 14-35　函数返回值使用实例

## 14.6.4　函数变量的作用域

函数体是完成特定功能的代码段，在代码执行过程总需要使用一些存放程序运行的中间结果的变量。变量分为局部变量和全局变量。局部变量是指在函数内部声明的变量，该变量只能在一段程序中发挥作用；而全局变量是指在函数之外声明的变量，该变量在整个 JavaScript 代码中发挥作用，全局变量的生命周期从声明开始，到页面关闭时结束。

局部变量和全局变量可以重名，也就是说，即便在函数体外声明了一个变量，在函数体内还可以再声明一个同名的变量。在函数体内部，局部变量的优先级高于全局变量，即在函数体内，同名的全局变量被隐藏了。

需要注意的是，专用于函数体内部的变量一定要用 var 关键字声明，否则该变量将被定义成全局变量，如果函数体外部有同名的变量，可能导致该全局变量被修改。

**【例 14-6-10】**变量的作用域范围。代码如下所示，页面效果如图 14-36 所示。


视频讲解

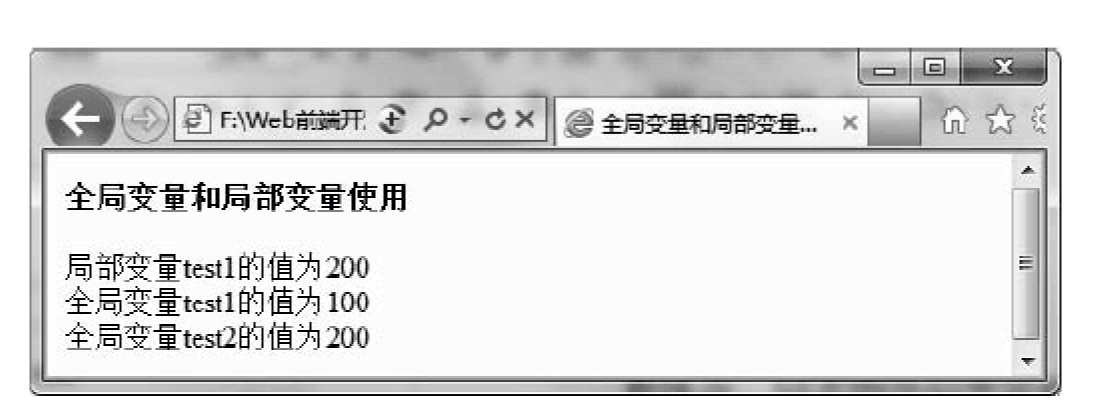



图 14-36　全局变量和局部变量使用实例

```
 1 <!-- edu_14_6_10.html -->
 2 <!doctype html>
 3 <html lang="en">
 4     <head>
 5         <meta charset="UTF-8">
 6         <title>全局变量和局部变量使用实例</title>
 7     </head>
 8     <body>
 9         <h4>全局变量和局部变量使用</h4>
10         <script type="text/javascript">
11             var test1 =100,test2 =100;     //定义全局变量
12             function checkScope( )
13             {
14                 var test1 =200;            //定义局部变量
15                 test2 =200;                //给全局变量再次赋值
16                 document.write("局部变量test1的值为"+test1);
17                 document.write("<br/>");
18             }
19             checkScope( );
20             document.write("全局变量test1的值为"+test1);
21             document.write("<br/>");
22             document.write("全局变量test2的值为"+test2);
23         </script>
24     </body>
25 </html>
```

**代码解释**

代码中第 11 行声明两个全局变量 test1 和 test2，初值均为 100；第 14 行用 var 声明了一个同名的局部变量 test1，初值为 200；第 15 行的全局变量 test2 赋值为 200(但没有用 var 声明)。由于局部变量和全局变量重名，根据规则，局部变量优先级更高，因此第 16 行输出显示结果是 200；第 20 行的 test1 是 100；第 22 行的 test2 也是全局变量，但是其值在第 15 行被修改为 200，结果显示为 200。

# 14.7　综 合 实 例

编程实现“手机批发业务-产品选购”页面，主要功能有查看购物车、收银台结算、初始化参数等。代码如下所示，效果如图 14-37 所示。其中，第一行的三件产品已经被选购。

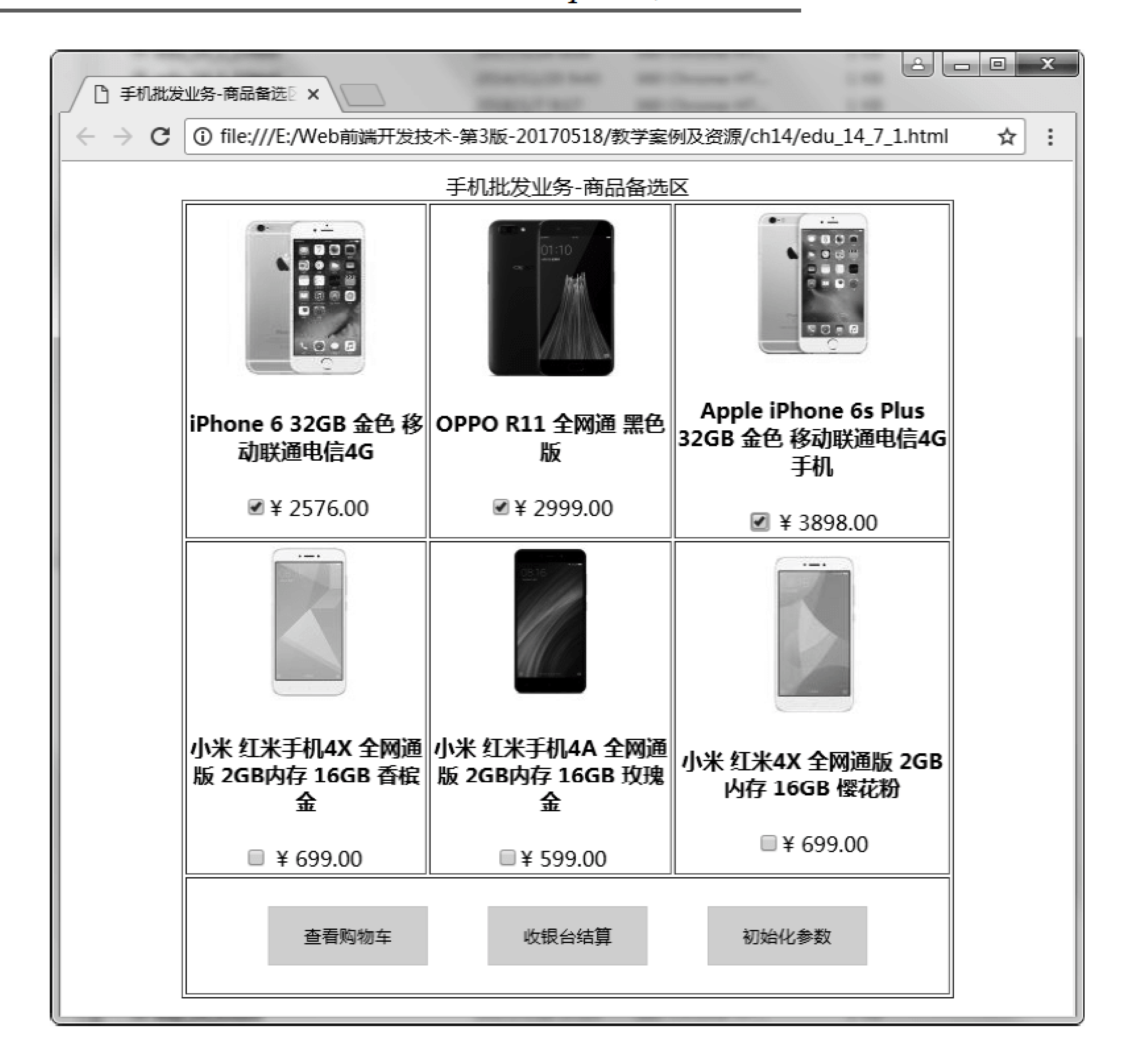



图 14-37　手机批发业务选购产品页面（已勾选 3 件）

根据如图 14-37 所示的页面效果，采用表单嵌套表格来进行页面布局。表格为 3 行 3 列，主要用于显示手机外形、参数和价格。

三个按钮的功能分别如下：“查看购物车”按钮的功能是当用户勾选相关产品后，能够显示用户所有选购信息，未勾选任何产品，提示告警信息，如图 14-38 所示。对应函数为 shoppingCart()，见代码中第 39～49 行。“收银台结算”按钮功能是当用户查看完购物车后，可以统计所选产品的件数和金额，通知支付，如图 14-39 所示。checkOut()，见代码中第 28～38 行。“初始化参数”按钮功能是将所有复选框变为未选中状态及数组清零。对应函数为 clearAll()，见代码中第 18～27 行。checkSelect(number)函数功能是检查页面上每个复选框的状态。除此之外，代码中第 15～17 行定义三个数组变量，分别用于商品价格、保存产品名称和商品选择状态。

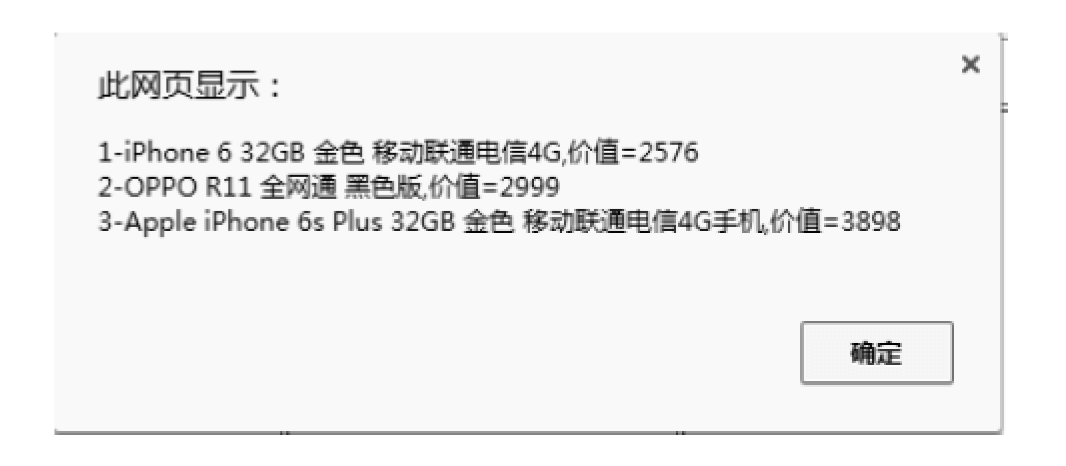



图 14-38　选择“查看购物车”对话框图

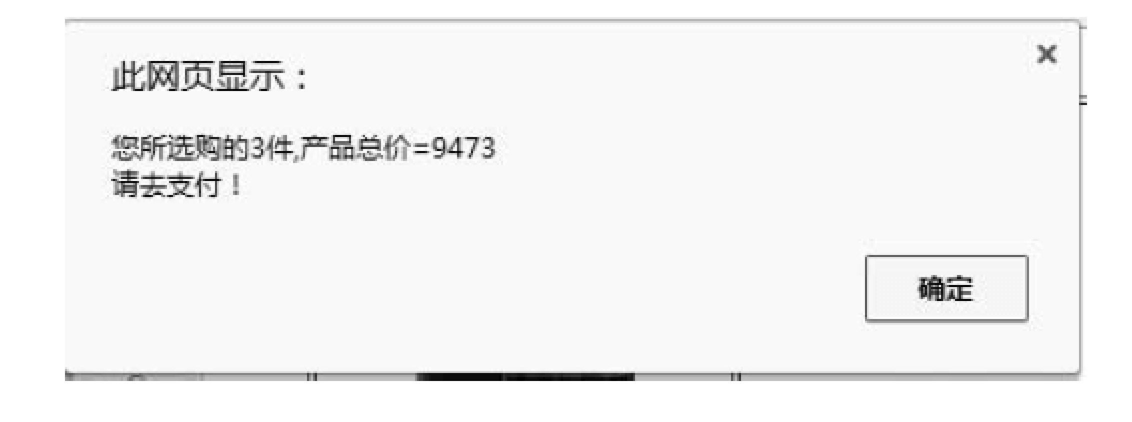



图 14-39　选择“收银台结算”对话框图

```
1 <!-- edu_14_7_1.html -->
2 <!doctype html>
3 <html lang="en">
4     <head>
```

```
        <meta charset="UTF-8">
        <title>手机批发业务-商品备选区</title>
        <style type="text/css">
            table {width: 580px; height: 200px;}
            td { text-align: center; vertical-align: middle;}
            .myBtn {margin: 20px; width: 120px; height: 45px;
                    border: 1px ridge #44FFEE;}
        </style>
        <script type="text/javascript">
            var result = ""; //存放选购信息
            var price = new Array(2576.00, 2999.00, 3898.00, 699.00, 599.00,
        699.00);
            var product = new Array("iPhone 6 32GB 金色 移动联通电信4G",
            "OPPO R11 全网通 黑色版", "Apple iPhone 6s Plus 32GB 金色 移动
            联通电信4G手机", "小米 红米手机4X 全网通版 2GB内存 16GB 香槟金", "
            小米 红米手机4A 全网通版 2GB内存 16GB 玫瑰金", "小米 红米4X 全网通
            版 2GB内存 16GB 樱花粉");
            var isSelected = new Array(0, 0, 0, 0, 0, 0);
            function clearAll() {
                isSelected = [0, 0, 0, 0, 0, 0]; //选择状态全部置0
                //所有复选框状态变为未选中状态
                myForm.sp0.checked = false;
                myForm.sp1.checked = false;
                myForm.sp2.checked = false;
                myForm.sp3.checked = false;
                myForm.sp4.checked = false;
                myForm.sp5.checked = false;
            }
            function checkOut() {
                var total = 0;//存放小计金额
                var count = 0;//存放选购产品件数
                for(var i = 0; i < isSelected.length; i++) {
                    count += isSelected[i];
                }
                for(var i = 0; i < price.length; i++) {
                    total = total + price[i] * isSelected[i]//累计金额
                }
               alert("您所选购的" + count + "件,产品总价=" + total+"\n"+"
               请去支付! ");
            }
            function shoppingCart() {
                //判断有多少个复选框被选中
                var selectList = ""; //保存所选产品清单
                for(var j = 0; j < product.length; j++) {
                 if(isSelected[j]) {//分行显示
                   selectList += (j + 1) + "-" + product[j] + ",价值=" +
                   price[j] + "\n";
                 }
                }
                var info = (selectList == "") ? "您的购物车为空，请选购! " :
                selectList;
                alert(info);//生成一个结算清单，显示输出
            }
            function checkSelect(number) {
                var temp;//暂存复选框状态
                switch(number) {
                    case 0:
```

```
                    temp = myForm.sp0.checked;break;
                case 1:
                    temp = myForm.sp1.checked; break;
                case 2:
                    temp = myForm.sp2.checked; break;
                case 3:
                    temp = myForm.sp3.checked; break;
                case 4:
                    temp = myForm.sp4.checked; break;
                default:
                    temp = myForm.sp5.checked; break;
            }
            isSelected[number] = (temp) ? 1 : 0; //记录下选中产品，1-选中，0-未选
        }
    </script>
</head>
<body>
    <form name="myForm" method="post" action="">
        <table align="center" border="1">
            <caption>手机批发业务-商品备选区</caption>
            <tr>
                <td><img src="mobile_1.jpg" /><br />
                    <h4 name="h41">iPhone 6 32GB 金色 移动联通电信4G</h4> <input type="checkbox" name="sp0" value="2576" onclick="checkSelect(0);">￥ 2576.00<br /></td>
                <td><img src="moblie_2.jpg" /><br />
                    <h4 name="h421">OPPO R11 全网通 黑色版</h4>
                    <input type="checkbox" name="sp1" value="2999" onclick="checkSelect(1);">￥ 2999.00<br /></td>
                <td><img src="moblie_3.jpg" /><br />
                    <h4 name="h43">Apple iPhone 6s Plus 32GB 金色 移动联通电信4G手机</h4>
                    <input type="checkbox" name="sp2" onclick="checkSelect(2);"> ￥ 3898.00<br /></td>
            </tr>
            <tr>
                <td><img src="moblie_4.jpg" /><br />
                    <h4 name="h44">小米 红米手机4X 全网通版 2GB内存 16GB 香槟金</h4> <input type="checkbox" name="sp3" value="699" onclick="checkSelect(3);"> ￥ 699.00
                    <br /></td>
                <td><img src="moblie_5.jpg" /><br />
                    <h4 name="h45"> 小米 红米手机4A 全网通版 2GB内存 16GB 玫瑰金</h4>
                    <input type="checkbox" name="sp4" value="599" onclick="checkSelect(4);">￥ 599.00<br /></td>
                <td><img src="moblie_6.jpg" /><br />
                    <h4 name="h46">小米 红米4X 全网通版 2GB内存 16GB 樱花粉</h4>
                    <input type="checkbox" name="sp5" value="699" onclick="checkSelect(5);">￥ 699.00<br /></td>
            </tr>
            <tr>
                <td colspan="3">
                    <input class="myBtn" type="button" value="查看购物车" onclick="shoppingCart();">
                    <input class="myBtn" type="button" value="收银台结
```

```
                            算" onclick="checkOut();">
                            <input class="myBtn" type="button" value="初始化参
                            数" onclick="clearAll();">
                        </td>
                    </tr>
                </table>
            </form>
        </body>
    </html>
```

# 本 章 小 结

JavaScript 是一种功能强大、使用简便、具有安全性的客户端脚本语言。本章简要地介绍了 JavaScript 语言的历史和特点，详细讲解了 JavaScript 的标识符、变量、运算符和表达式、三种程序控制结构（包括顺序结构、分支结构和循环结构）及函数等相关知识。通过在 HTML 文档中嵌入 JavaScript 脚本语言，可以增强用户与网页之间的交互性，并在页面中实现各种特效，提高页面的观赏性。

# 练习与实验

## 练习 14

**1．选择题**

（1）在客户端网页脚本语言中最为通用的是（　　）。

A．JavaScript　　B．VB　　C．Perl　　D．ASP

（2）下列不是 JavaScript 的特点的是（　　）。

A．跨平台性　　B．动态性　　C．编译型语言　　D．解释型语言

（3）下列不属于 JavaScript 的关键字的是（　　）。

A．for　　B．interface　　C．switch　　D．new

（4）下列属于 JavaScript 常量的是（　　）。

A．NaN　　B．undfined　　C．Math.PI　　D．Infinity

（5）JavaScript 中表示声明变量的关键字是（　　）。

A．if　　B．while　　C．var　　D．loop

（6）下列定义函数 display()语法正确的是（　　）。

A．function display( ){ }　　B．function: display( ){ }

C．function=display( ){ }　　D．display( ){ }

（7）引用外部 show.js 文件方法正确的选项是（　　）。

A．<script src="show" ></script>　　B．<script name="show.js" ></script>

C．<script href="show.js" ></script>　　D．<script src="show.js" ></script>

**2．填空题**

（1）在 HTML 中嵌入 JavaScript 代码时，需使用________标记。

（2）JavaScript 中的消息对话框分为________、________和________三种。

（3）表达式 18/0 的值为________。

（4）逻辑表达式(5<100) &&(3>0)的结果为________。

（5）位表达式 5 & 7、5 | 7 和 5 ^ 7 的结果分别为________、________和________。

**3．简答题**

（1）continue 与 break 语句在循环中的作用有什么不同？

（2）do…while()与 while 循环有什么不同？

（3）自定义函数时应注意哪些事项？

## 实验 14

1．编写 JavaScript 程序实现“九九乘法口诀”表，如图 14-40 所示。

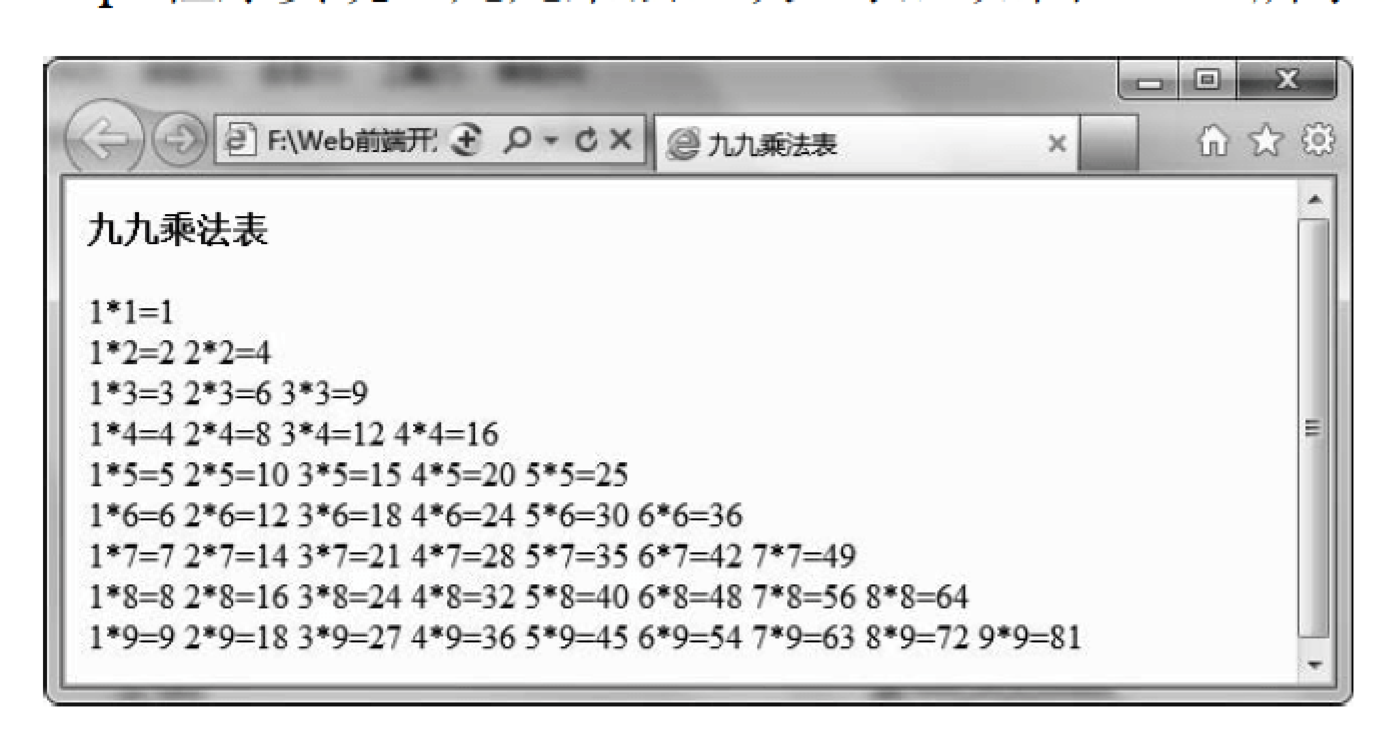



图 14-40　九九乘法表效果图

2．编写 JavaScript 代码，找出符合条件的数。如图 14-41 所示。

（1）页面标题为：“找出符合条件的数”；

（2）页面内容：3 号标题标记显示“找出 1000～9999 之间能够被 17 和 13 同时整除的整数的个数及累加和”，要求输出区间累计有多少个符合条件的整数，并计算符合条件的整数的累加和，同时输出符合条件的整数，输出格式：每行 10 个整数。




图 14-41　找出符合条件的数

# 第 15 章　JavaScript 事件分析

本章学习目标

通过 JavaScript 事件知识的学习，能够了解网页中基本的事件类型，理解 JavaScript 事件在网页设计中的作用。理解事件及事件句柄的相关概念；掌握 JavaScript 中常用的事件句柄；理解事件发生时的事件处理过程。

Web 前端开发工程师应掌握以下内容：

- 了解 JavaScript 事件类型。
- 理解事件的概念。
- 理解事件句柄与事件处理代码相关联的方式。
- 学会利用表单的提交及重置事件对表单的数据进行校验。
- 理解鼠标事件中的鼠标单、双击及鼠标移动事件。
- 掌握常用的键盘及窗口事件。

## 15.1　JavaScript 事件概述

事件是一些可以通过脚本响应的页面动作。当用户按下鼠标键或者提交一个表单，甚至在页面上移动鼠标时，就会产生相关的事件。绝大多数事件的命名是描述性的，很容易理解，例如 Click、Submit、MouseOver 等，通过名称就可以猜测其含义。

### 15.1.1　事件类型

JavaScript 中的事件大多数与 HTML 标记相关，都是由用户操作页面元素时触发的。根据事件触发的来源及作用对象的不同，可把事件分为鼠标事件、键盘事件、HTML 事件及突变事件 4 种类型。

**1．鼠标事件**

鼠标事件主要指用户使用鼠标操作 HTML 元素时触发的事件。常见的鼠标单击、双击、文本框选择、单选按钮、复选框选中都会触发鼠标事件。当鼠标移动、盘旋、移出网页上相关区域内的特定元素时触发 MouseMove、MouseOver 和 MoveOut 事件。

**2．键盘事件**

键盘事件主要指用户在键盘上敲击、输入时触发的事件。如用户在键盘上按下某一键时会触发 KeyDown 事件，用户释放按下的键会触发 KeyUp 事件。

**3．HTML 事件**

HTML 事件主要指当窗口发生变动或者发生特定的客户端/服务器端交互时触发的事

件。如页面完全载入时在 window 对象上会触发 Load 事件；任何元素或者窗口本身失去焦点时触发 Blur 事件。

**4．突变事件**

突变事件主要指文档对象底层元素发生改变时触发的事件。如当文档或者元素的子树因为添加或者删除节点而改变时会触发 DomSubtreeModified（DOM 子树修改）事件；当一个节点作为另一个节点的子节点插入时会触发 DomNodeInserted（DOM 节点插入）事件。

## 15.1.2 事件句柄

事件句柄（又称事件处理函数）是指事件发生时要进行的操作。每一个事件均对应一个事件句柄，在程序执行时，将相应的函数或语句指定给事件句柄，则在该事件发生时，浏览器便执行指定的函数或语句，从而实现网页内容与用户操作的交互。当浏览器检测到某事件发生时，便查找该事件对应的事件句柄有没有被赋值，如果有，则执行该事件句柄。通常，事件句柄的命名原则是在事件名称前加上前缀 on。如鼠标移动 MouseOver 事件，其事件句柄为 onMouseOver。事件句柄名称与 HTML 标记的事件处理属性相同。

**1．基本语法**

```
<标记  事件句柄="JavaScript代码">… </标记>
 <input type="button" name="" value="显示" onclick="show();">
```

**2．语法说明**

事件句柄名称与事件属性同名，都作为 HTML 标记的属性，与事件名称略有不同，只是在事件名称前面加上了 on。例如 Click 事件的事件句柄为 onClick，该项标记对应的事件属性也为 onClick；Blur 事件的事件句柄为 onBlur，该项标记对应的事件属性也为 onBlur，其他事件的事件句柄以此类推。常用的事件和事件句柄的对照关系如表 15-1 所示。

**表 15-1　事件类型、事件、事件句柄一览表**

| 事件类型 | 事　件 | 事件句柄 | 事件解释 |
|---|---|---|---|
| 键盘事件 | KeyDown | onKeyDown | 当键盘被按下时执行 JS 代码 |
| | KeyPress | onKeyPress | 当键盘被按下后又松开时执行 JS 代码 |
| | KeyUp | onKeyUp | 当键盘被松开时执行 JS 代码 |
| 鼠标事件 | Click | onClick | 当鼠标被单击时执行 JS 代码 |
| | Dblclick | onDblclick | 当鼠标被双击时执行 JS 代码 |
| | MouseDown | onMouseDown | 当鼠标按钮被按下时执行 JS 代码 |
| | MouseMove | onMouseMove | 当鼠标指针移动时执行 JS 代码 |
| | MouseOut | onMouseOut | 当鼠标指针移出某元素时执行 JS 代码 |
| | MouseOver | onMouseOver | 当鼠标指针悬停于某元素之上时执行 JS 代码 |
| | MouseUp | onMouseUp | 当鼠标按钮被松开时执行 JS 代码 |
| 表单控件事件 | Change | onChange | 当元素改变时执行 JS 代码 |
| | Submit | onSubmit | 当表单被提交时执行 JS 代码 |
| | Reset | onReset | 当表单被重置时执行 JS 代码 |
| | Select | onSelect | 当元素被选取时执行 JS 代码 |
| | Blur | onBlur | 当元素失去焦点时执行 JS 代码 |
| | Focus | onFocus | 当元素获得焦点时执行 JS 代码 |

续表

| 事件类型 | 事 件 | 事件句柄 | 事件解释 |
| --- | --- | --- | --- |
| 窗口事件 | Load | onLoad | 当文档载入时执行 JS 代码 |
| | UnLoad | onUnload | 当文档卸载时执行 JS 代码 |

### 15.1.3 事件处理

只要给特定的事件句柄绑定事件处理代码就可以响应事件。事件处理指定方式有 3 种：在 HTML 标记中的静态指定、在 JavaScript 中的动态指定及特定对象的特定事件的指定。

**1. 静态指定**

1）基本语法

```
<标记 事件句柄1="事件处理程序1" [事件句柄2= "事件处理程序2"  ...  事件句柄n="事件处理程序n"]>…</标记>
```

2）语法说明

静态指定方式，是在开始标记中设置相关事件句柄，并绑定事件处理程序即可。一个标记可以设置一个或多个事件句柄，并绑定事件处理程序。事件程序可以是 JavaScript 代码串或函数，通常将事件处理程序定义成函数。

例如，给 p 标记和 body 标记添加事件句柄属性，并绑定事件。格式如下：

```
<p onClick="show();" onDblClick="display();"></p>
<body onLoad="alert('页面装载成功！');" onUnload="pageLoad();"></body>
```

**【例 15-1-1】**在 HTML 标记中的静态指定事件处理代码。代码如下所示，页面效果如图 15-1 和图 15-2 所示。

视频讲解

```
<!-- edu_15_1_1.html -->
<!doctype html>
<html lang="en">
   <head>
       <meta charset="UTF-8">
       <title>HTML属性的事件处理器的应用</title>
       <script type="text/javascript">
           function testInfo(message){alert(message);}
       </script>
   </head>
   <body>
       <h4>HTML属性的事件处理器的应用</h4>
       <form method="post" action="">
           <input type="button" value="通过JS语句输出信息"  onclick="alert
           ('使用alert()输出信息')">
           <input type="button" value="通过函数输出信息" onclick="testInfo
           ('调用testInfo()函数输出信息')">
       </form>
   </body>
</html>
```

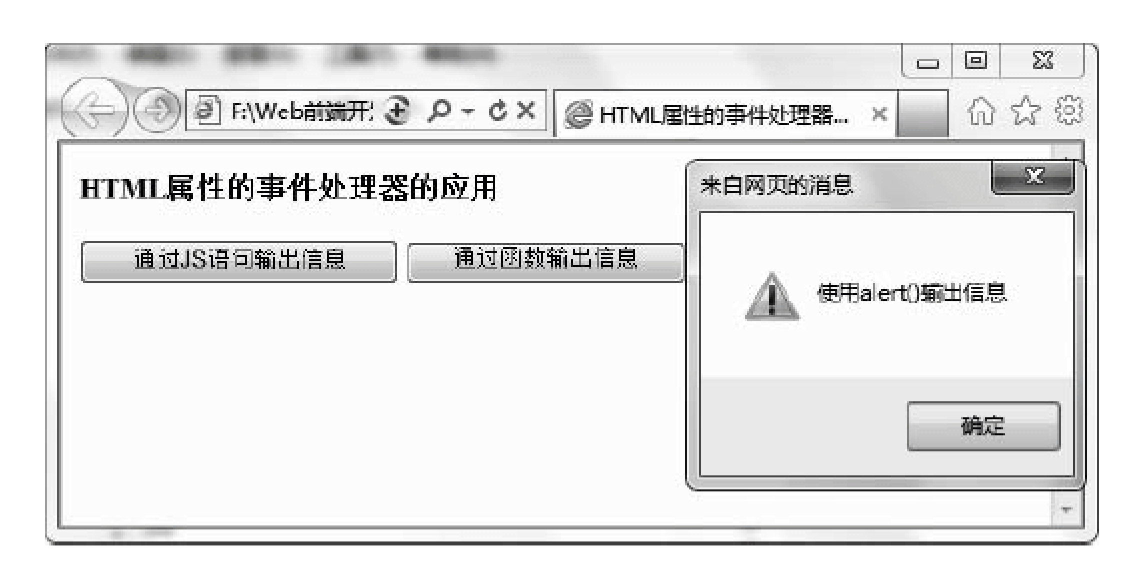



图 15-1 调用 JS 语句输出信息




图 15-2 调用函数输出信息

3）代码解释

代码中第 14 行、第 15 行定义了 2 个普通按钮，并通过 HTML 的 input 标记的 onClick 事件句柄来关联事件处理程序。如果单击“通过 JS 语句输出信息”按钮，将触发该按钮的 Click 事件，直接执行 JavaScript 代码 alert('使用 alert()输出信息')，弹出告警消息框，显示信息；如果单击“通过函数输出信息”按钮，将触发该按钮的 Click 事件，调用代码中第 7～9 行定义的名为 testInfo(message)函数，通过参数传递要输出的信息，函数的执行结果是弹出告警消息框，并把参数传递的信息显示在对话框内。

**2．动态指定**

大多数情况下使用静态指定方式来处理事件，但有时也需要在程序运行过程中动态指定事件，也称为分配某一事件，这种方式允许程序像操作 JavaScript 属性一样来处理事件。

1）基本语法

```
<事件源对象>.<事件句柄>=function(){<事件处理程序>;}
Object.onclick=function(){disp();}
                                        //动态给对象指派事件，绑定事件处理函数
Object.onclick();                       //调用方法
```

2）语法说明

在此用法中，“事件处理程序”必须使用不带函数名的 function(){}来定义，也就是无函数名的函数，函数体内可以是字符串形式的代码，也可以是函数。

**【例 15-1-2】**在 JavaScript 中进行动态指定事件处理程序。代码如下所示，页面效果如图 15-3 所示。

视频讲解

```
<!-- edu_15_1_2.html -->
<!doctype html>
<html lang="en">
    <head>
        <meta charset="UTF-8">
        <title> JavaScript中的动态指定</title>
        <style type="text/css">
            #inp{width:100px;height:40px;color:red;}
        </style>
        <script type="text/javascript">
            function clickHandler()
            {
                alert("代码触发事件，即将提交表单！");
                return true;
            }
        </script>
```

```
17     </head>
18     <body>
19         <form name="myform" method="post" action="" >
20             <input id="inp" type="button" name="mybutton" value="提交" >
21         </form>
22         <script type="text/javascript">
23             //向 button 元素动态分配 onclick 事件
24             document.getElementById('inp').onclick=function(){return
               clickHandler();}
25             myform.mybutton.onclick();     //程序触发
26         </script>
27     </body>
28 </html>
```

3）代码解释

代码中第 11～15 行定义了一个名为 clickHandler()的函数；第 19～21 行定义了一个表单，并在表单中插入一个按钮，第 22～26 行插入脚本，其中第 24 行通过 JavaScript 程序给 button 按钮动态分配 onclock 事件，当代码执行时，第 25 行系统自动执行了动态分配的 onclick 事件，采用名称调用按钮的 onclick 事件时必须在事件属性后面加上一对小括号，例如 obj.onclick()，否则调用无效，而不是用户单击“提交”按钮的结果。




图 15-3　JavaScript 动态指定处理事件函数

### 3．特定对象特定事件的指定

在 script 标记中编写元素对象的事件处理程序代码。使用 script 标记的 for 属性指定事件源，并用 event 属性指定事件句柄名称，这种方法用得比较少，但是在某些场合还是很好用的。

1）基本语法

```
<script type="text/javascript"  for="对象"  event="事件句柄">
    //事件处理程序代码
</script>
```

2）语法说明

for 属性指定特定对象，如 window、document 等；event 属性指定事件句柄名称，如 onload、onunload 等。在脚本 script 标记中插入相关事件处理函数代码。

**【例 15-1-3】**特定对象的特定事件处理程序的应用，代码如下所示，页面效果如图 15-4 所示。

视频讲解

```
1 <!-- edu_15_1_3.html -->
2 <!doctype html>
3 <html lang="en">
4    <head>
5       <meta charset="UTF-8">
6       <title>给特定对象指定特定事件处理程序</title>
7    </head>
8    <body>
9       <h4>给特定对象指定特定事件处理程序</h4>
10      <script type="text/javascript" for="window" event="onload">
11         alert("网页读取完成，欢迎光临！");
12      </script>
13   </body>
14 </html>
```

3）代码解释

代码中第 10～12 行利用 script 标记的 for、event 属性分别指定对象和事件句柄，并在脚本中定义了 JavaScript 代码 alert("网页读取完成，欢迎光临！")。当网页加载时，通过告警消息框输出“网页读取完成，欢迎光临！”。




图 15-4　特定对象的特定事件处理程序

## 15.1.4　事件处理程序的返回值

在 JavaScript 中通常事件处理程序不需要有返回值，这时浏览器会按默认方式进行处理。很多情况下需要使用返回值，来判断事件处理程序是否正确进行处理，或者通过这个返回值来判断是否进行下一步操作。

在这种情况下，事件处理程序返回值都为布尔型值，如果为 false，则阻止浏览器的下一步操作；如果为 true，则进行默认的操作。

**1．基本语法**

```
<标记 事件句柄="return 函数名（参数）;" >…</标记>
```

**2．语法说明**

事件处理代码中函数必须具有布尔型的返回值，即函数体中最后一句必须是带返回值的 return 语句。

【例 15-1-4】事件处理程序返回值的应用。代码如下所示，页面效果如图 15-5 所示。

视频讲解

```
1 <!-- edu_15_1_4.html -->
2 <!doctype html>
3 <html lang="en">
4    <head>
5       <meta charset="UTF-8">
6       <title>事件处理程序返回值的应用</title>
7       <script language="javascript">
8          function showName(){
9             if(document.form1.name1.value=="")
10            {  alert("没有输入内容！"); return false; }
11            else {
12               alert("欢迎你！"+document.form1.name1.value);return true;
13            }
14         }
15      </script>
16   </head>
17   <body>
18      <h4>事件处理程序返回值的应用</h4>
19      <!-- onsubmit事件处理函数返回真值就执行action指定的网页 -->
20      <form name="form1" action="simple.html" onsubmit="return showName();">
21         姓名：<input type="text" name="name1" />
22         <input type="submit" value="提交"/>
23      </form>
24   </body>
25 </html>
```

### 3. 代码解释

代码中第 8～14 行定义一个 JavaScript 函数 showName()；如果表单中姓名文本输入框中没有内容，则提示“没有输入内容！”，返回 false 值；如果姓名文本框中输入姓名，如“储久良”，则提示“欢迎你！储久良”，单击“确定”按钮后，返回 true 值。第 20～23 行定义一个表单，并在表单中插入一个文本输入框和一个提交按钮，其中第 20 行定义了表单的 onsubmit 事件句柄，并指定事件发生时调用执行代码“return showName();”。

浏览网页，并在姓名文本框处输入姓名，单击“提交”按钮，触发 Submit 事件，调用执行代码“return showName();”，返回值为 true，则浏览器进行下一步操作，访问网页 simple.html，如图 15-6 所示；如果在文本输入框中不输入任何内容就单击“提交”按钮则返回 false 值，浏览器阻止进行下一步操作，返回输入界面。




图 15-5　事件处理程序的返回值应用

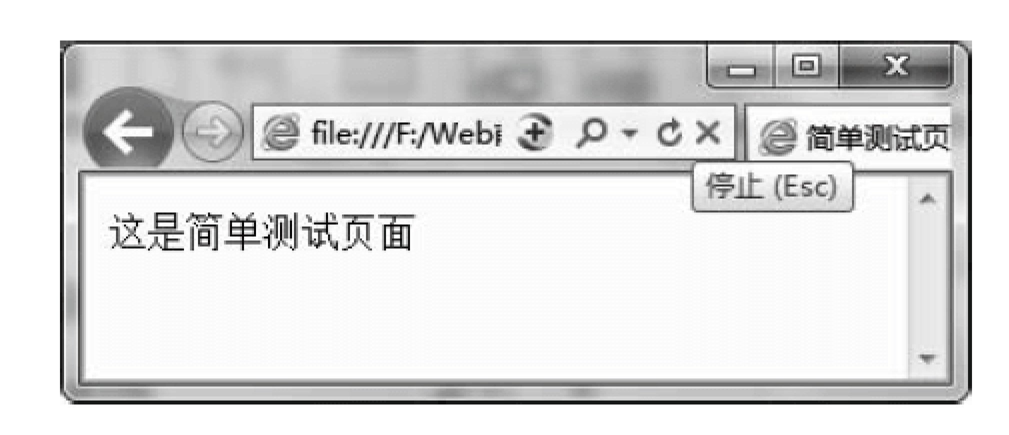



图 15-6　返回值为真时的网页

# 15.2 表 单 事 件

表单是 Web 应用中和用户进行交互的最常用的工具。用户注册、发表讨论和评论等都需要用到表单。用户在表单中填写数据，然后将数据发送到服务器端。JavaScript 脚本所要做的主要工作就是表单验证，如验证用户是否有未填信息，输入的数据格式是否正确等。这样，在数据被提交到服务器之前数据的正确性和合法性就得到了验证并反馈给了用户，用户可以根据提示修改错误。

表单控件（元素）有很多，如文本输入框、下拉列表框、复选框、单选按钮、提交按钮等。在对表单控件（元素）进行操作时，都会触发相应的事件。

## 15.2.1 获得焦点与失去焦点事件

当表单控件获得焦点时会触发 focus 获得焦点事件，当表单控件失去焦点时会触发 blur 事件。当单击表单中的按钮时，该按钮就获得了焦点；当单击表单中的其他区域时，该按钮就失去了焦点。

**【例 15-2-1】**表单控件焦点事件的应用。代码如下所示，页面效果如图 15-7 所示。

```
1 <!-- edu_15_2_1.html -->
2 <!doctype html>
3 <html lang="en">
4     <head>
5         <meta charset="UTF-8">
6         <title>获得/失去焦点测试</title>
7         <script type="text/javascript">
8             function getFocus(){document.bgColor ="#114455";}
9             function loseFocus(){document.bgColor ="#FF66FF";}
10        </script>
11    </head>
12    <body>
13        <form>
14            <br/><input type="button" onfocus="getFocus()" value="获得/
              失去焦点触发事件" onblur="loseFocus()"/>
15        </form>
16    </body>
17 </html>
```

视频讲解



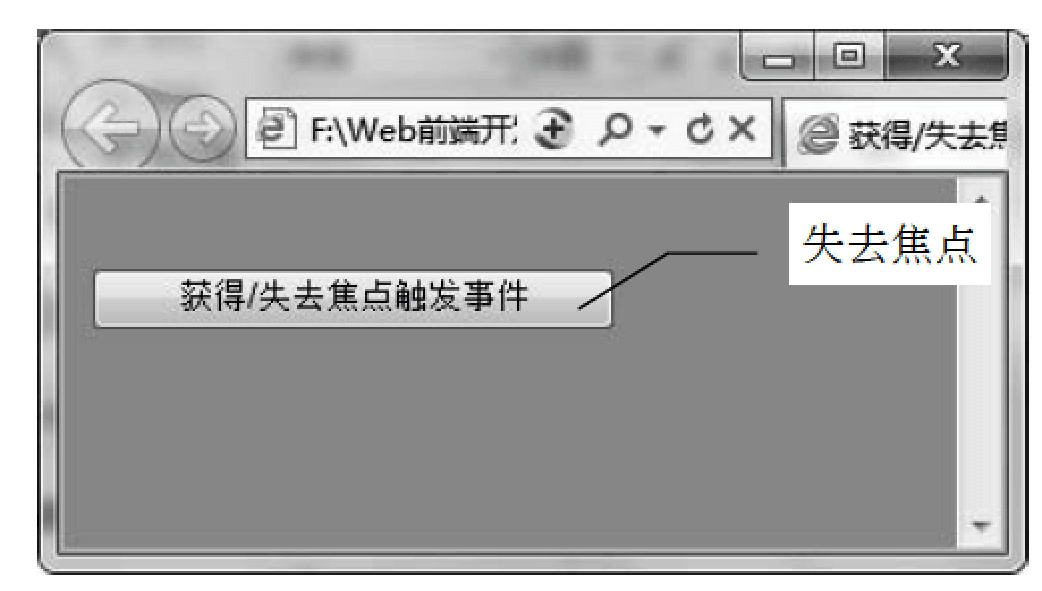




图 15-7　普通按钮获得/失去焦点效果图

**代码解释**

代码中第 14 行定义了“获得/失去焦点触发事件”的普通按钮，并为此按钮设置了

onFocus 和 onBlur 事件句柄，当该按钮获得焦点时会触发获得焦点事件，调用 getFocus() 函数将文档背景颜色设置为#114455，当该按钮失去焦点时会触发失去焦点事件，调用 loseFocus()函数将文档背景颜色设置为#FF66FF。

## 15.2.2 提交及重置事件

在表单中单击“提交”按钮后，会触发 Submit 事件，将表单中的数据提交到服务器端；当单击“重置”按钮后，会触发 Reset 事件，将表单中的数据重置为初始值。在表单中，插入 1 个 type 属性值为 submit 的 input 标记来添加 1 个提交按钮，当单击该按钮时会触发表单的 Submit 事件；同样可以插入 1 个 type 属性值为 reset 的 input 标记来添加 1 个重置按钮，当单击该按钮时会触发表单的 Reset 事件。如果需要表单 Submit 事件及 Reset 事件触发时完成特定的功能，例如需要对表单数据进行合法性验证，则需要为表单设置事件句柄，并自定义相关函数。

**【例 15-2-2】**表单提交、重置事件的应用。代码如下所示，页面效果如图 15-8 所示。

视频讲解

```
<!-- edu_15_2_2.html -->
<!doctype html>
<html lang="en">
   <head>
      <meta charset="UTF-8">
      <title>表单提交、重置事件的应用</title>
      <style type="text/css">
         fieldset{width:300px;height:150px;}
      </style>
      <script language="javascript" type="text/javascript">
         function $(id){return document.getElementById(id);}
         function submitTest(){
            var msg ="用户名:"+$("input1").value;
            msg+="\n密码:是"+$("input2").value;
            alert(msg);
            return false;
         }
         function resetTest(){alert("将数据清空");}
      </script>
   </head>
   <body>
      <form onSubmit="return submitTest();"  onReset="resetTest()">
         <fieldset>
            <legend>表单数据提交</legend>
            <br><label>用户名：</label><input type="text" id="input1">
            <br><label>密 码：</label><input type="password"
            id="input2">
            <br><input type="submit" value="提交">
            <input type="reset" value="重置">
         </fieldset>
      </form>
   </body>
</html>
```




图 15-8　单击“提交”按钮时的提示信息

**代码解释**

代码中第 10～19 行定义了三个 JavaScript 函数，分别是$(id)、submitTest()和 resetTest()；第 22 行为表单设置了 onSubmit 和 onReset 事件句柄。

当单击“提交”按钮将表单数据提交时，触发 Submit 事件，调用执行代码“return submitTest();”，这段代码将调用 submitTest()，获取输入框中的用户名和密码，弹出告警消息框并返回 false；当单击“重置”按钮将表单数据重置时，触发 Reset 事件，调用执行代码“resetTest()”，这段代码调用执行后会弹出“数据清空”的告警消息框。

## 15.2.3　改变及选择事件

在表单中，当选择了文本输入框或多行文本输入框内的文字时会触发 Select 选择事件。部分示例代码如下所示：

```
<form>
<input type="text" name="" value="文本被选择后触发事件" onSelect=
"Javascript:alert('内容已被选中!')">
</form>
```

代码中第 2 行定义了一个文本输入框，并设置 onSelect 属性值为 JavaScript 代码；当文本框的内容被选中后，将触发 Select 事件，调用代码，通过告警消息框弹出一个显示“内容已被选中!”的对话框；当一个文本输入框或多行文本输入框失去焦点并更改值时或当 select 下拉选项中的一个选项状态改变后会触发 Change 改变事件。

【例 15-2-3】下拉列表框实现图像切换。代码如下所示，页面效果如图 15-9 所示。

视频讲解

```
<!-- edu_15_2_3.html -->
<!doctype html>
<html lang="en">
    <head>
        <meta charset="UTF-8">
        <title>下拉菜单</title>
        <script language="javascript">
            function $(id){return document.getElementById(id);}//获取元素
            function changeImage(){
                var index =$("game").selectedIndex;//获取下拉框中选择项
                $("show").src=$("game").options[index].value;// 更改图片
            }
        </script>
    </head>
```

```
15     <body>
16         <div align="center">
17             <form >
18                 <select id="game" onChange="changeImage()" >
19                     <option value="pic4.jpg">--请选择--</option>
20                     <option value="pic0.jpg">平板电视</option>
21                     <option value="pic1.jpg">笔记本电脑</option>
22                     <option value="pic2.jpg">单反相机</option>
23                     <option value="pic3.jpg">智能手机</option>
24                 </select>
25             </form>
26         </div>
27         <p align="center">
28             <img  src="pic4.jpg"  id="show">
29         </p>
30     </body>
31 </html>
```

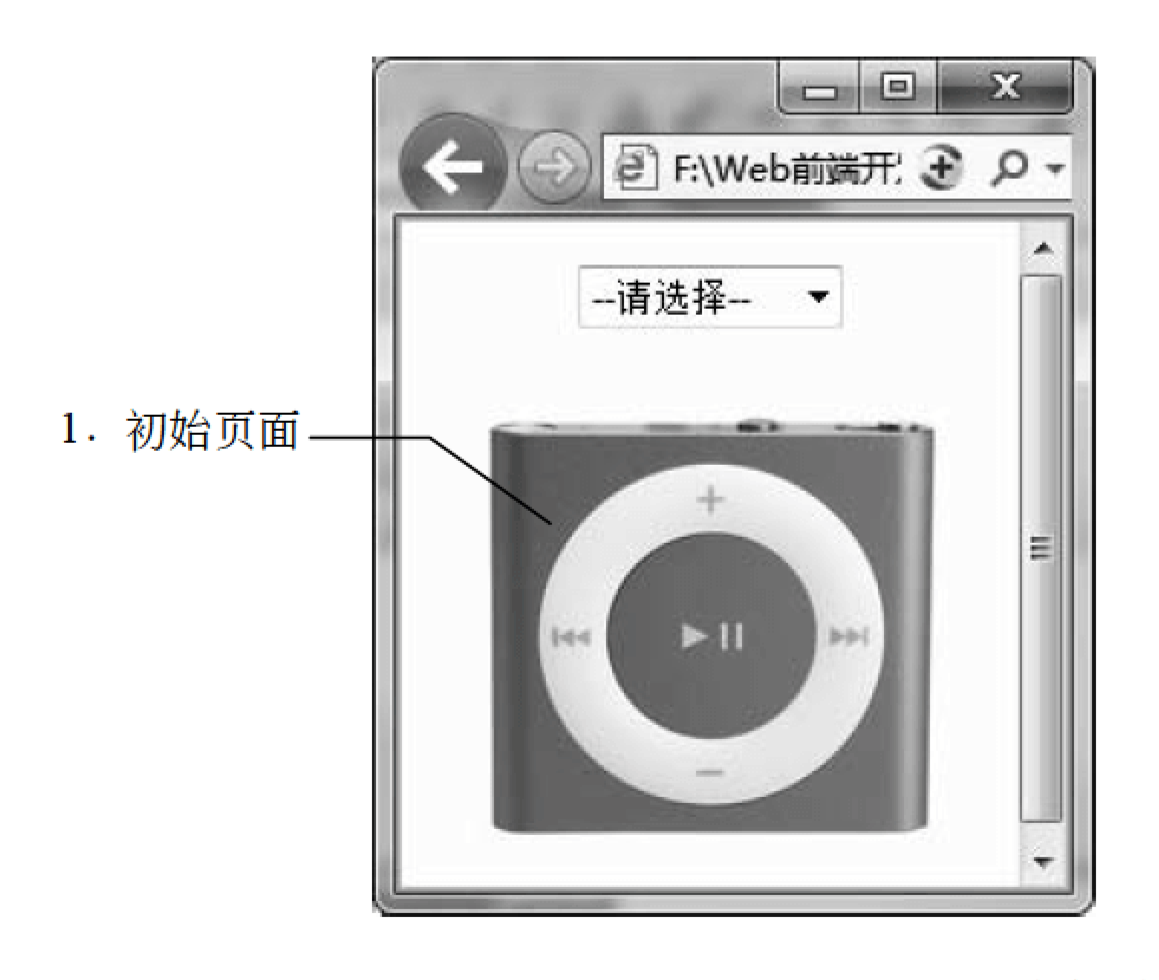





图 15-9　下拉列表框选择“单反相机”选项

**代码解释**

代码中第 8～12 行定义了两个 JavaScript 函数，分别为$(id)、changeImage()；第 18 行为下拉列表框设置了 onChange 事件句柄；第 28 行在页面中插入了一张图像。当下拉列表框中的选项改变时会触发 change 事件，调用 changeImage()将原来的图像更改为选中的图像。

changeImage()中第 10 行用于获得下拉列表框中选中的列表项的索引；第 11 行用于将指定索引处的下拉列表框选项的值赋给 img 元素的 src 属性来完成图片的更换。

## 15.3　鼠 标 事 件

在网页设计中，如果用鼠标对网页中控件进行操作时会触发鼠标事件。当单击鼠标左键会触发 Click 事件，双击鼠标时会触发 DblClick 事件，鼠标按下后再松开时会触发 MouseUp 事件等。下面对一些常用的鼠标事件做简单介绍。

## 15.3.1　鼠标单、双击事件

鼠标事件主要指用鼠标对页面中的控件进行单击或双击操作时触发的事件，它们也是网页开发中运用最多的事件。当单击页面中的按钮时可以触发鼠标单击事件，例如：

```
<input type="button" name="click" value="鼠标单击" onClick="alert('你单击了我 !')">
```

当双击页面中的按钮时可以触发鼠标双击事件，例如：

```
<input type="button" name="click"  value="鼠标双击" onDblClick="alert('你双击了我 !')">
```

【例 15-3-1】文本输入框内容复制。代码如下所示，页面效果如图 15-10 所示。

视频讲解

```
1 <!-- edu_15_3_1.html -->
2 <!doctype html>
3 <html lang="en">
4     <head>
5         <meta charset="UTF-8">
6         <title>鼠标单击事件</title>
7         <script type="text/javascript">
8             function $(id){return document.getElementById(id);}
9             function copyText(){$("target").value=$("source").value;}
10        </script>
11    </head>
12    <body>
13        <h4>文本框内容复制</h4>
14        <form method="post" action="">
15            来源文本框：<input type="text" id="source" value=""><br>
16            目标文本框：<input type="text" id="target" readonly><br>
17            <input type="button" value="复制文本框内容" onClick=
              "copyText();">
18        </form>
19    </body>
20 </html>
```

**代码解释**

代码中第 8 行、第 9 行定义了两个函数，分别为$(id)、copyText()；第 15 行、第 16 行定义了两个文本输入框，且第 2 个文本框设置只读属性；第 17 行定义了一个普通按钮，并为该按钮设置了 onCilck 事件句柄。在第 1 个文本输入框中输完内容后，单击“复制文本框内容”按钮时会触发 Click 事件，调用 copyText()函数完成文本框内容的复制。

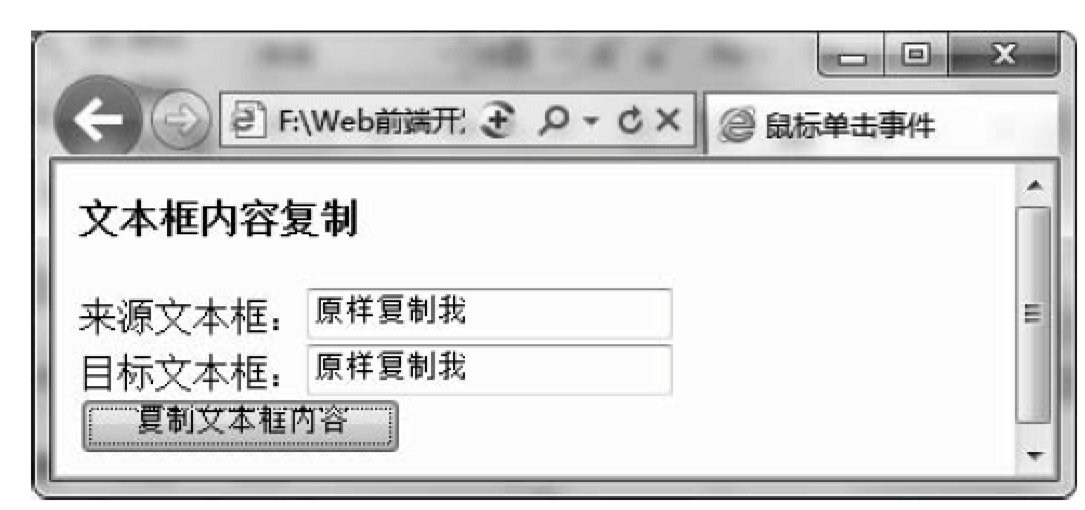



图 15-10　鼠标单击事件

## 15.3.2　鼠标移动事件

鼠标事件除了最典型的 Click 事件之外，还有鼠标进入页面元素 MouseOver 事件、退出页面元素 MouseOut 事件和鼠标按键检测 MouseDown 及 MouseUp 等事件。下面的例子实现了鼠标移向页面中的某个图像时触发 MouseOver 事件，鼠标移出图像时触发了 MouseOut 事件。

**【例 15-3-2】**移动鼠标替换图片。代码如下所示，页面效果如图 15-11 所示。

视频讲解

```
1 <!-- edu_15_3_2.html -->
2 <!doctype html>
3 <html lang="en">
4     <head>
5         <meta charset="UTF-8">
6         <title>鼠标移动事件</title>
7         <script type="text/javascript">
8             function $(id){return document.getElementById(id);}
9             function mouseOver(){$('b1').src ="eg_mouse1.jpg";}
10            function mouseOut(){$('b1').src ="eg_mouse2.jpg";}
11        </script>
12        <style type="text/css">
13            p,h4{text-align:center;}
14        </style>
15    </head>
16    <body>
17        <h4>鼠标事件</h4>
18        <hr color="blue">
19        <p>
20            <img alt="鼠标移动事件" src="eg_mouse2.jpg" id="b1"
              onmouseover="mouseOver()" onmouseout="mouseOut()"/>
21        </p>
22    </body>
23 </html>
```

**代码解释**

代码中第 8～10 行定义了三个 JavaScript 函数，分别为$(id)、mouseOver()和 mouseOut()。第 20 行定义了一个图像并为该图像设置了 onMouseOver 和 onMouseOut 事件句柄。当鼠标移到图像区域时会触发 MouseOver 事件执行 mouseOver()，将当前图像更换为新的图像，如图 15-11（a）所示；当鼠标移出图像区域时会触发 MouseOut 事件执行 mouseOut()，将当前图像恢复为原来的图像，如图 15-11（b）所示。




（a）鼠标移入的页面




（b）鼠标移出后的页面

图 15-11　鼠标移动事件的应用

## 15.4 键盘事件




键盘事件主要有 3 个，分别是 KeyDown、KeyPress 及 KeyUp 事件，它们用来检测键盘按下、按下松开及完全松开这些动作。通过 window 对象的 event 对象中的 event.keyCode 可以获得按键对应的键码值。

**【例 15-4-1】**键盘事件的应用。代码如下所示，页面效果如图 15-12 所示。

```
<!--  edu_15_4_1.html -->
<!doctype html>
<html lang="en">
    <head>
        <meta charset="UTF-8">
        <title>键盘事件的应用</title>
        <script type="text/javascript">
            function checkNo()
            {
                if(window.event.keyCode!=13)
                {
              if(event.keyCode<48 || event.keyCode>57){alert("你输入学号错误！");}
                }else{
                    if(myform.sno.value.length<=0){alert("学号不能为空");}
                    else{alert("你的学号为："+myform.sno.value);}
                }
            }
            function checkName(){
                if(window.event.keyCode==13){alert("你的姓名为："+myform.sname.value);}
            }
        </script>
    </head>
    <body>
        <h4>键盘事件的应用</h4>
        <form name="myform" method="post" action="">
            学号：<input type="text" name="sno" id="sno" onKeyPress="checkNo()">必须为数字<br>
            姓名：<input type="text" name="sname" id="sname" onkeypress="checkName()">回车显示姓名<br>
            <input type="submit" value="提交"><input type="reset">
        </form>
    </body>
</html>
```

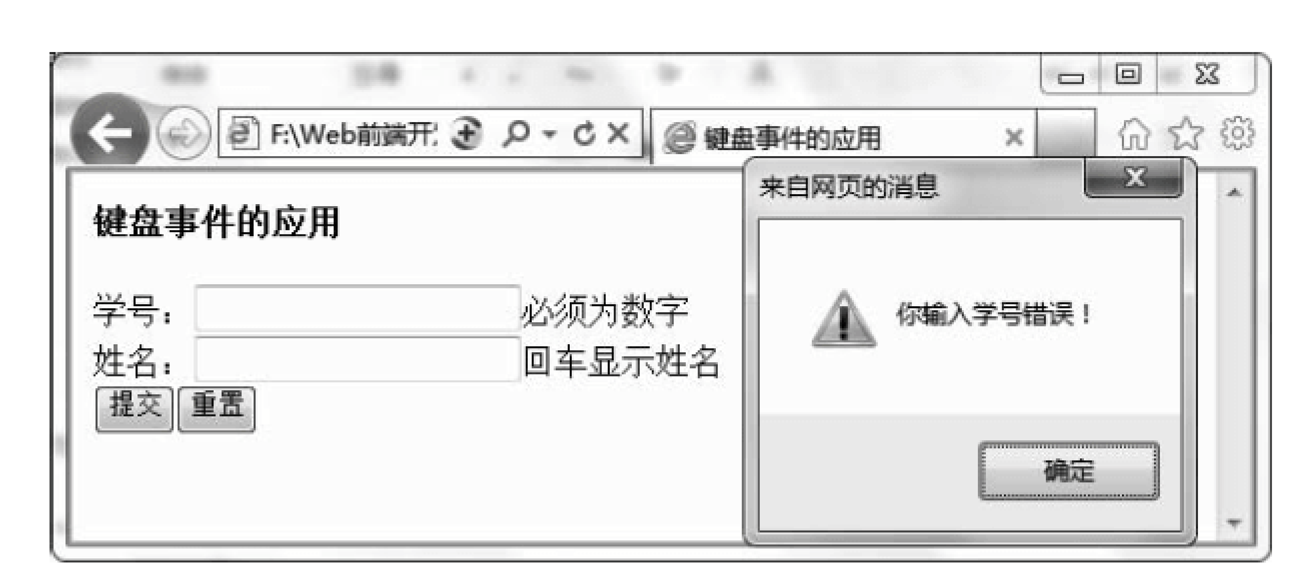



图 15-12　键盘事件的应用

**代码解释**

代码中第 7～21 行定义了两个 JavaScript 函数，分别是 checkNo()及 checkName()。第 25～29 行定义了一个表单，在表单中插入两个文本输入框，并为两个文本输入框设置了 onKeyPress 事件句柄。在 sno 文本输入框中通过键盘输入学号时，则会触发 KeyPress 事件调用 checkNo()执行检查，如果键盘按下的不是 Enter 键且输入的不是数字时（Enter 键的键码值是 13，数字键 0～9 对应的键码值 48～57），则给出“你输入学号错误！”的提示信息；如果用户按下的是 Enter 键且输入的是数字键，则给出“你的学号为：XXX”的提示信息；如果用户没有输入数据直接按下 Enter 键，则给出“学号不能为空”的提示信息。在 sname 文本输入框中输入姓名时，则会触发 KeyPress 事件调用 checkName()函数执行检查，如果按下的是 Enter 键则给出“你的姓名为：XXX”的提示信息。

## 15.5 窗 口 事 件

窗口事件是指浏览器窗口在加载页面或卸载页面时触发的事件。加载页面时会触发 Load 事件，卸载页面时会触发 UnLoad 事件，这两个事件与<body>及<frameset>两个页面元素有关。

**【例 15-5-1】**窗口事件的应用。代码如下所示，页面效果如图 15-13 所示。



视频讲解

```
1 <!-- edu_15_5_1.html -->
2 <!doctype html>
3 <html lang="en">
4    <head>
5        <meta charset="UTF-8">
6        <title>窗口事件的应用</title>
7        <script type="text/javascript">
8            function load(){alert("欢迎访问本页面!");}
9            function unload(){alert("欢迎下次访问!");}
10       </script>
11   </head>
12   <body onload="load();" onunload="unload();"
13       <h4>窗口事件的应用</h4>
14       <p onclick="alert('单击我! ')">单击我! </p>
15   </body>
16 </html>
```

**代码解释**

代码中第 7～10 行定义了两个 JavaScript 函数，分别是 load()、unload()函数。第 12 行为该页面的 body 元素设置了 onLoad 及 onUnLoad 事件句柄。当浏览器窗口加载该页面时会触发 Load 事件调用 load()函数，弹出“欢迎访问本页面!”的提示信息，如图 15-13（a）所示；当单击段落时，触发 Click 事件，弹出“单击我！”告警消息框，如图 15-13（b）所示；当关闭该浏览器窗口或当前页面跳转到其他页面时会触发 unLoad 事件，调用 unload()函数，弹出“欢迎下次访问!”的提示信息，如图 15-13（c）所示。




（a）加载页面时效果图　（b）单击时效果图　（c）离开页面时效果图

图 15-13　加载事件/卸载事件页面效果图

# 15.6　综 合 实 例

在网页设计与开发过程中，经常利用表单提交与重置事件对表单中数据进行验证。例如进入当当网的注册界面，输入数据完成后单击“用户注册”按钮，如果数据格式符合要求则将数据提交到服务器端，显示注册成功；如果输入的邮箱/手机号码或密码格式不正确时，则要求重新输入。

用 JavaScript 程序也可以模拟一个注册过程，当单击注册按钮时，如果验证合法，则将数据提交；否则，继续保持登录页，并给出相关提示信息。

**【例 15-6-1】**用户注册信息验证。代码如下所示，页面效果如图 15-14 所示。




视频讲解

图 15-14　单击“用户注册”按钮验证页面效果图

```
<!--  edu_15_6_1.html -->
<!doctype html>
<html lang="en">
    <head>
        <meta charset="UTF-8">
        <title>用户注册页面</title>
        <style type="text/css">
            strong{color:red;font-style:bolder;}
            fieldset{width:560px;height:186px;padding:0px 50px;}
            #button{margin:10px 20px;}
        </style>
        <script  type="text/javascript">
            function $(id){return document.getElementById(id);}
            function checkReg()
            {
                var username=$("myname").value;
```

```
            var pwd=$("mypwd1").value;
            var pwdConfirm=$("mypwd2").value;
            var checkright=true;
            if(username=="" || pwd=="")  //两者中有一个为空
            {
                alert("请确认用户名和密码输入是否正确！！");
                checkright=false;
            }else   //不为空，再判断用户名和密码的长度合法性
            {
                if(username.length<6)
                {
                    alert("用户名长度太短，至少6个字符！！");
                    checkright=false;
                }else if(pwd.length<6){
                    alert("密码长度太短，至少6个字符！！");
                    checkright=false;
                }else if(pwd!=pwdConfirm){
                    alert("两次输入的密码必须一致！！");
                    checkright=false;
                }else{
                    checkright=true;}
                }
                return checkright;
            }
            function clearInfo()
            {
                var flag = confirm("确认要重置数据吗？");
                if(flag==true)
                {
                    $("myname").value = "";
                    $("mypwd1").value = "";
                    $("mypwd2").value = "";
            }
        }
    </script>
</head>
<body>
    <form action="regsuccess.html" method="get" onSubmit="return checkReg()" onReset="clearInfo()">
        <fieldset>
            <legend align="center" >新用户注册</legend><br>
            <div>
                <label >用 户 名：</label>
                    <input type="text" name="myname" id="myname">
                    <strong>(用户名要大于6位)</strong><br>
                    <label> 登录密码：</label>
                    <input type="password" name="mypwd1" id="mypwd1">
                    <strong>(密码要大于6位)</strong><br>
                    <label> 密码确认：</label>
                    <input type="password" name="mypwd2" id="mypwd2">
                    <br>
                     <input id="button" type="submit" value="用户注册" >
                <input  id="button" type="reset" value="重置">
            </div>
        </fieldset>
    </form>
```

```
69     </body>
70 </html>
```

**代码解释**

代码中第 12～51 行定义了三个 JavaScript 函数，分别是$(id)、checkReg()、clearInfo() 函数。第 54 行为表单设置了 onSubmit 和 onReset 事件句柄。当单击“用户注册”按钮将表单数据提交时，此时会触发 Submit 事件，调用执行代码 return checkReg()，首先判断用户输入的用户名称或密码是否为空，如果为空则弹出提示对话框，并返回 false；接着判断用户名或密码是否大于六个字符，如果不是则返回 false；最后判断两次输入的密码是否相同，如果不同则返回 false；其他情况返回 true。

如果注册用户名少于六个字符，单击“用户注册”按钮后会弹出“用户名长度太短，至少 6 个字符！！”的提示信息，如图 15-14 所示。

当单击“重置”按钮将表单数据重置时，此时会触发 Reset，调用执行函数 clearInfo( )，该函数的作用是提醒用户是否将表单数据重置，如图 15-15 所示。




图 15-15　单击“重置”按钮时页面效果图

# 本 章 小 结

本章介绍 JavaScript 脚本中的事件处理的概念、方法，列出了常用的事件及事件句柄，并且介绍了如何编写用户自定义的事件处理函数以及如何将它们与页面中用户的动作相关联，以得到预期的交互性能。

重点介绍了 Web 开发中常用的表单事件、鼠标事件、键盘事件等。在表单事件中，详细介绍表单元素的焦点事件、表单提交与重置事件以及表单元素的选中及改变事件。在鼠标事件中，详细介绍鼠标单击及鼠标移动事件。在窗口事件中，主要介绍了装载事件和卸载事件。Web 前端开发人员只要掌握 JavaScript 事件概念、事件触发类型和事件处理的方式，就可以开发出具有交互性、动态性的页面。

# 练习与实验

## 练习 15

**1．选择题**

（1）以下选项中，鼠标单击事件对应的事件句柄是（　　）。

A．onChange　　B．onLoad　　C．onClick　　D．onDblclick

（2）以下事件中，当页面中的文本输入框获得焦点时触发的事件是（　　）。

A．click　　B．load　　C．blur　　D．focus

（3）以下事件中，表单数据填完后，单击提交按钮，会触发的事件是（　　）。

A．submit　　B．reset　　C．click　　D．focus

（4）以下选项中，表单重置事件对应的事件句柄是（　　）。

A．onSubmit　　B．onReset　　C．onChange　　D．onLoad

（5）以下选项中，将 validate()函数和一个按钮的单击事件关联起来正确的用法是（　　）。

A．<input type="button " value="校验" onClick="validate( ) ">

B．<input type="button " value="校验" onDbClick="validate( ) ">

C．<input type="button " value="校验" onSubmit="validate( ) ">

D．<input type="button " value="校验" onReset="validate( ) ">

（6）以下事件中，不属于键盘事件的是（　　）。

A．KeyDown　　B．KeyPress　　C．KeyUp　　D．KeyOver

（7）JavaScript 中的 Load 事件的作用是（　　）。

A．浏览器窗口加载页面时，执行的 JavaScript 事件

B．浏览器窗口离开页面时，执行的 JavaScript 事件

C．用户提交一个表单时，执行的 JavaScript 事件

D．鼠标移出对象时，执行的 JavaScript 事件

**2．填空题**

（1）事件句柄的命名原则是在事件名称前加上前缀________。

（2）JavaScript 中事件处理方式可以有三种，分别是________、__________、________。

（3）当表单中的表单控件获得焦点时会触发__________事件，该事件对应的事件句柄为__________；表单数据提交时会触发__________事件，该事件对应的事件句柄为__________。

（4）鼠标单击时会触发__________事件，浏览器窗口在加载页面时会触发__________事件。

**3．简答题**

（1）网页开发中常见的事件类型有哪些？分别代表什么操作？

（2）事件发生时，对事件的处理方式有哪几种？

（3）表单事件中最常用的事件有哪些？举例说明它们在实际开发中的应用。

## 实验 15

1．编写 JavaScript 代码实现用户登录时数据合法性校验功能，界面如图 15-16 所示。具体要求如下：




图 15-16　用户登录页面

（1）必填项验证——用户名文本输入框、密码输入框必须含有值。

（2）有效性验证——用户名、密码长度大于等于 8 个字符，小于等于 20 个字符。

（3）当提交数据时，如果输入框中的数据不合法，则给出对应的提示信息并将焦点聚焦到对应的输入框上。

提示：使用域和域标题进行窗口布局，背景颜色为#663399。

2．编写 JavaScript 程序实现单击列表框任一选项时，通过告警消息框显示教材名称及定价，如图 15-17 所示。

（1）页面标题为“显示列表项的内容”;

（2）页面内容：3 号标题标记显示标题“显示列表项的内容”；插入一个大小为 5 的列表框，用于显示教材名称，教材定价保存在列表项的 value 中，分别如下：

计算机组成原理 35 元、数据结构 38 元、计算机网络 43 元、Java 程序设计 40 元、算法分析 28 元。

（3）编写 displayItem()函数，实现当用户选择某一列表项时通过告警框分行显示选中的教材名称和定价（列表项的内容和 value 值）。

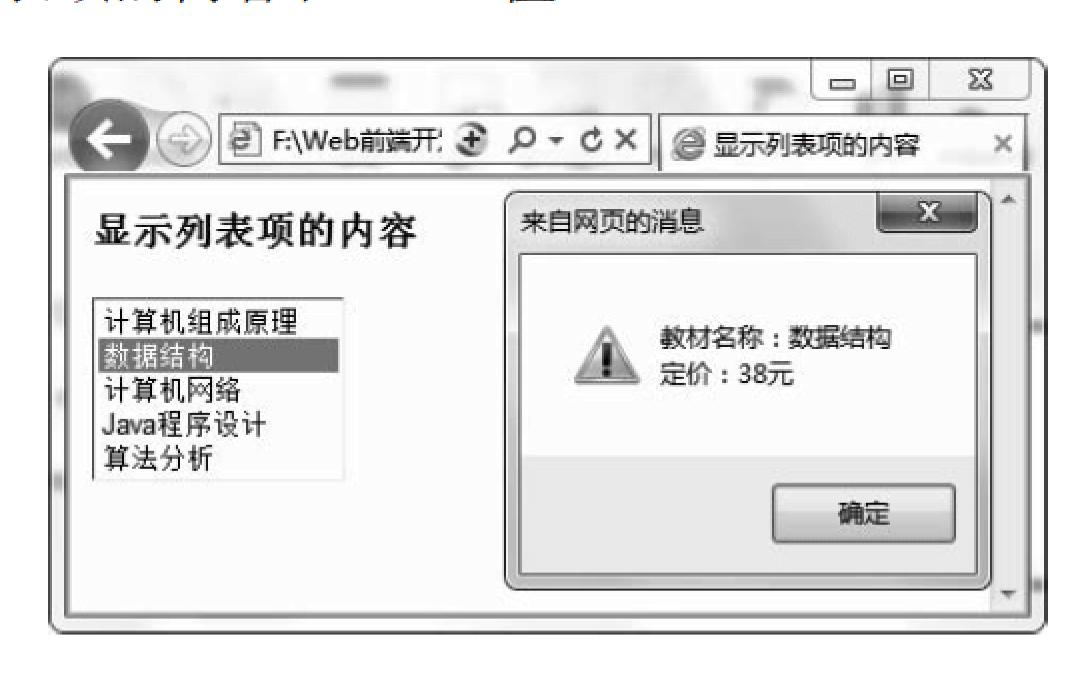



图 15-17　显示列表项内容

# 第 16 章　DOM 和 BOM

**本章学习目标**

JavaScript 既支持传统的结构化编程，同时也支持面向对象的编程，用户在编程时可以使用 JavaScript 语言提供的不同类型的对象，也可以自己定义对象的类型。一个完整的 JavaScript 实现是由三个不同部分组成的，分别是核心（ECMAScript）、文档对象模型（Document Object Model，DOM）及浏览器对象模型（Browser Object Model，BOM）。通过本章的学习，能够掌握 JavaScript 语言中内置对象的常用属性及方法，理解 DOM 及 BOM 的概念。掌握运用 document 对象来访问、创建及修改节点；掌握 window 对象的常用属性及方法；了解 navigator、screen、history、location 等对象。

Web 前端开发工程师应掌握以下内容：

- 学会使用 JavaScript 内置对象的常用属性及方法。
- 理解文档对象模型的节点树的构建及节点类型的划分。
- 学会使用 document 对象常用方法来设计动态效果的网页。
- 理解浏览器对象模型的各对象的层次关系。
- 学会使用 window 对象的定时器及对话框方法。
- 了解 navigator、screen、history、location 等对象的属性和方法。

## 16.1　JavaScript 常用对象

JavaScript 对象是拥有属性和方法的数据。采用面向对象编程能够减轻编程人员的工作量，提高设计 Web 页面的能力。

JavaScript 的对象类型可以分为四类：

（1）本地对象(native object)，ECMA-262 定义为“独立于宿主环境的 ECMAScript 实现提供的对象”。简单来说，本地对象就是 ECMA-262 定义的类（引用类型），包括 Object、Function、Array、String、Boolean、Number、Date、RegExp、Error、EvalError、RangeError、ReferenceError、SyntaxError、TypeError、URIError 等。这些对象独立于宿主环境，先定义对象，实例化后再通过对象名来使用。

（2）内置对象(built-in object)。由 ECMAScript 实现提供的、不依赖于宿主环境的对象，在 ECMAScript 运行之前就已经创建好的对象就叫作内置对象。这意味着开发者不必明确实例化内置对象，因为它已被实例化了。ECMA-262 只定义了两个内置对象，即 Global 和 Math。Global 是全局对象，全局对象只是一个对象，而不是类。既没有构造函数，也无法实例化一个新的全局对象。例如 isNam()、isFinite()、parseInt()和 parseFloat()等，都是 Global

对象的方法。Math 对象可直接使用，如 Math.Random()、Math.round(20.5)等。

（3）宿主对象(host object)。ECMAScript 实现的宿主环境提供的对象。所有 BOM 和 DOM 对象都是宿主对象。通过它可以与文档和浏览器环境进行交互，如 document、window 和 frames 等。

（4）自定义对象。根据程序设计需要，由编程人员自行定义的对象。例如定义一个 person 对象，它有四个属性分别是 firstName、lastName、age、eyeColor，同时给属性赋值。定义代码格式如下所示：

```
var person=new Object();   /* 这是一种方法*/
person.firstname="Bill";
person.lastname="Gates";
person.age=56;
person.eyecolor="blue";
var person={firstName:"John", lastName:"Doe", age:50, eyeColor:"blue"};
/*另一方法*/
```

在面向对象编程过程中，所有对象都必须先定义再实例化，然后才能使用。使用 new 运算符来创建对象，例如，“var obj=new Object();”。定义后使用对象的方法是“对象名称.方法名( )”；访问对象属性的方法是“对象名称.属性名”。JavaScript 中包含了一些常用的对象，如 Array、Boolean、Date、Math、Number、String、Object 等。这些对象常用在客户端和服务器端的 JavaScript 中，下面对这些常用对象做简单介绍。

### 16.1.1 Array

Array 对象用于在单个的变量中存储多个相同类型的值，其值可以是字符串、数值型、布尔型等，但由于 JavaScript 是弱类型的脚本语言，所以数组元素也可以不一致。通过声明一个数组，将相关的数据存入数组，使用循环等结构对数组中的每个元素进行操作。

作为 Web 前端开发人员，在编程时应尽量保证数组中的元素数据类型相同，这是一种良好的编程习惯。

**1．创建 Array 对象**

1）基本语法

```
var stu1=new Array();
var stu2=new Array(size);
var stu3=new Array(element0, element1, ..., elementn);
```

2）参数说明

参数 size 定义数组元素的个数。返回的数组的长度 stu2.length 等于 size。

参数 element0，…，elementn 是参数列表。当使用这些参数来调用构造函数 Array()时，新创建的数组的元素就会被初始化为这些值。

**2．数组的返回值**

数组变量 stu1、stu2、stu3 返回新创建并被初始化了的数组。如果调用构造函数 Array()时没有使用参数，那么返回的数组为空，数组的 length 为 0。当调用构造函数时只传递给它一个数字参数，该构造函数将返回具有指定个数、元素为 undefined 的数组。当其他参数调用 Array()时，该构造函数将用参数指定的值初始化数组。当把构造函数作为函数调用，

不使用 new 运算符时，它的行为与使用 new 运算符调用它时的行为完全一样。格式如下：

```
var stu = ["张有为","蒋丽娟","王一新","李大为"];
```

### 3. 数组元素初始化与修改指定数组元素

如果数组没有初始化，即是空数组时，可以使用循环给数组元素进行赋值，也可以一一赋值。如：stu[i] = 表达式，i 为 0～course.length-1 之间，也称为数组的下标。如果数组下标超出了数组的边界，则返回值为 undefined。可以用赋值的方式来修改数组对应位置的元素。代码如下所示：

```
var stu = new Array();          /*先定义数组*/
stu[0] = "王大为";               /*给数组元素赋值*/
stu[1] = "李永明";               /*给数组元素赋值*/
var len=stu.length;             /*len的值为2*/
stu[1] = "张慧娟";               /* 修改数组中第2个元素 */
```

### 4. 数组对象的属性和方法

Array 对象的长度可以通过 length 属性值来获取。Array 对象最常用的方法及说明如表 16-1 所示。

**表 16-1　Array 方法及说明**

| 方　法 | 说　明 |
|---|---|
| join（分隔符） | 把数组的所有元素放入一个字符串。元素通过指定的分隔符进行分隔 |
| pop() | 删除并返回数组的最后一个元素 |
| push（新元素） | 向数组的末尾添加一个或更多元素，并返回新的长度 |
| shift() | 删除并返回数组的第一个元素 |
| unshift（新元素） | 向数组的开头添加一个或更多元素，并返回新的长度 |
| sort() | 对数组的元素进行排序 |
| reverse() | 颠倒数组中元素的顺序 |
| splice() | 删除元素，并向数组添加新元素 |
| slice() | 从某个已有的数组返回选定的元素 |
| toString() | 把数组转换为字符串，并返回结果 |
| toLocaleString() | 把数组转换为本地数组，并返回结果 |
| concat() | 连接两个或更多的数组，并返回结果 |

**【例 16-1-1】**数组的属性和方法应用。代码如下所示，页面效果如图 16-1 所示。

```
<!-- edu_16_1_1.html -->
<!doctype html>
<html lang="en">
    <head>
        <meta charset="UTF-8">
        <title>数组对象的应用</title>
    </head>
    <body>
        <h3>数组对象的应用</h3>
        <script type="text/javascript">
            var stu1 = new Array("张有为","蒋丽娟","王一新","李大为");
            var stu2 =["张祥雨","姜进步","王新力","刘大山"];
```

视频讲解

```
13            document.write("数组中的元素：<br>");
14            //访问数组中的元素
15            for (var i=0;i<=stu1.length-1;i++ )
16            {
17                document.write(i+"-"+stu1[i]+"   ");
18            }
19            document.write("<br><br>");
20            //join方法的使用
21            document.write(stu2.join("-")+"<br>");//"-"分隔
22            document.write(stu2.join("+")+"<br>");//"+"分隔
23            document.write(stu2.join()+"<br>");    //默认
24            //pop,push方法的使用
25            document.write("<br>删除数组最后元素是"+stu2.pop());
26            var s=stu2.push("沈通达","高学衡");
27            document.write("<br>数组2的长="+s);
28            var stu1 = new Array ("张有为","蒋丽娟","王一新","李大为");
29            //shift,unshift方法的使用
30            var ss=stu1.shift();
31            document.write("<br>删除数组第一个元素是："+ss);
32            //在数组开始处插入新元素
33            var s=stu1.unshift("徐丽丽");            //在IE中显示undefined
34            document.write("<br>数组元素分别："+stu1+"<br>数组的长度="+s);
                                                   //在IE中用stu1.length代替
35        </script>
36    </body>
37 </html>
```

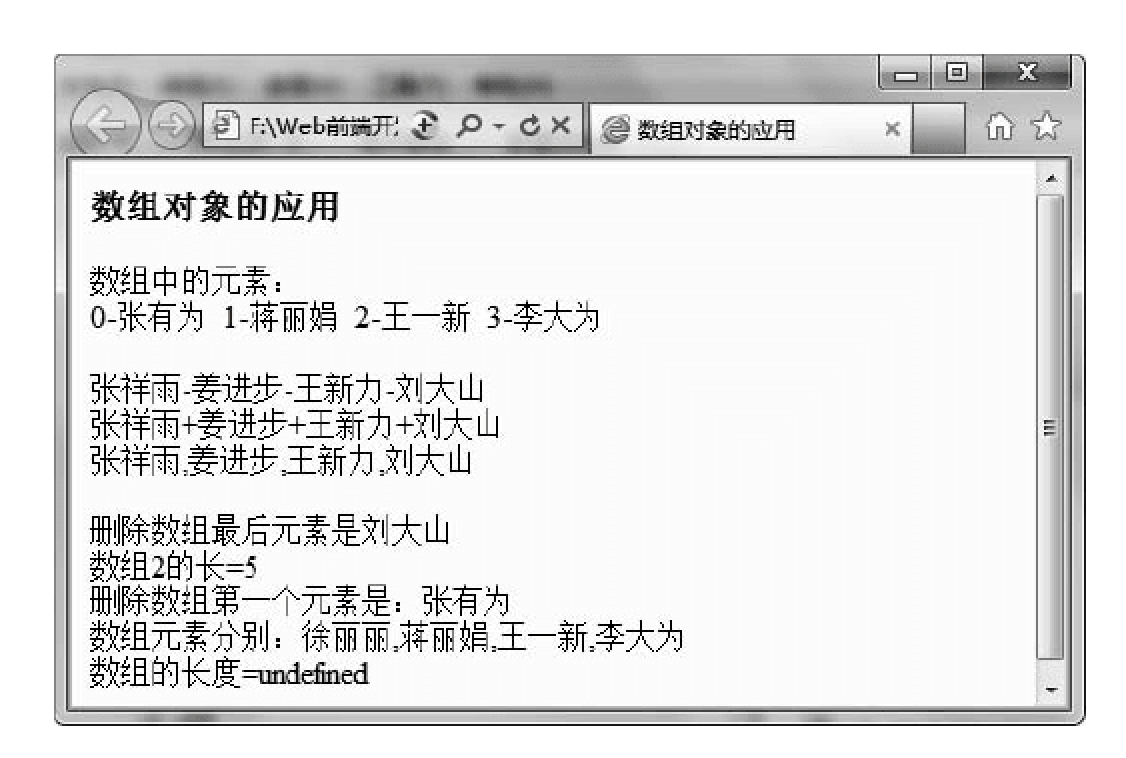



图 16-1　数组对象的应用

**代码解释**

代码中第 11 行、第 12 行定义了两个数组对象 stu1、stu2，并采用两种方法给数组赋值。第 13～34 行通过使用数组的属性及相关方法对数组进行遍历和修改。

注：在 IE 浏览器中，代码中第 33 行无法正常执行，即变量 s 未赋值，所以第 34 行中显示数组的长度为 undefined（未定义），而在其他浏览器中能够正常显示数组的长度。

## 16.1.2　Date

JavaScript 脚本核心对象 Date 用于处理日期和时间。Date 对象有很多方法，可以提取时间和日期。

### 1. 创建日期对象

基本语法：

```
var today=new Date();
var today=new Date(毫秒数);
var today=new Date(标准时间格式字符串);
var today=new Date(年,月,日,时,分,秒,毫秒);
```

根据上述创建方法，可以用下列格式来定义日期对象。格式如下：

```
var today=new Date();                          //自动使用当前的日期和时间
var today=new Date(3000);                      //1970年1月1日, 0时0分3秒
var today=new Date("Apr 15,2016 15:20:00");//2016年4月15日15时20分0秒
var today=new Date(2016,3,25,14,42,50,50); //2016年4月25日14时42分50秒
```

### 2. 日期对象的方法

日期对象中包含着丰富的信息，可以通过日期对象提供的一系列方法分项提取出年、月、日、时、分、秒等各种信息。Date 对象方法及说明如表 16-2 所示。

**表 16-2　提取日期对象每个字段的方法及说明**

| 方　法　名 | 说　　明 |
|---|---|
| getDate() | 从 Date 对象返回一个月中的某一天(1～31) |
| getDay() | 从 Date 对象返回一周中的某一天(0～6) |
| getMonth() | 从 Date 对象返回月份(0～11) |
| getFullYear() | 从 Date 对象以 4 位数字返回年份 |
| getHours() | 返回 Date 对象的小时数(0～23) |
| getMinutes() | 返回 Date 对象的分钟数(0～ 59) |
| getSeconds() | 返回 Date 对象的秒数(0～ 59 |
| getMilliseconds() | 返回 Date 对象的毫秒数(0～999) |
| getTime() | 返回 1970 年 1 月 1 日至今的毫秒数 |

【例 16-1-2】获得当前日期对象的年、月、日、时、分、秒，并且以特定的格式显示在页面中。代码如下所示，页面效果如图 16-2 所示。



视频讲解

```
<!--  edu_16_1_2.html -->
<!doctype html>
<html lang="en">
    <head>
        <meta charset="UTF-8">
        <title>日期对象的应用</title>
    </head>
    <body >
        <h4>日期对象的应用</h4>
        <script type="text/javascript">
            var now = new Date();
            var y = now.getFullYear();
            var m = now.getMonth()+1;
            var d = now.getDate();
            var h = now.getHours();
            var mi = now.getMinutes();
            var s = now.getSeconds();
            if(m<10){m="0"+m;}
```

```
19            if(d<10){d="0"+d;}
20            if(h<10){h="0"+h;}
21            if(mi<10){mi="0"+mi;}
22            s=(s<10)?("0"+s):s;  //if(s<10){s="0"+s;}
23            var str = y+"年"+m+"月"+d+"日 "+h+":"+mi+":"+s;
24            document.write(str);
25        </script>
26    </body>
27 </html>
```

**代码解释**

代码中第 11 行定义了一个日期对象 now，代表了当前的日期时间。第 12～17 行调用对象 now 的相关方法将该对象的年、月、日、小时、分钟、秒获取并显示在页面上。




图 16-2　显示当前系统日期和时间

需要注意的是，日期中的 1～12 月用数字 0～11 表示；每周的星期日～星期六，用数字 0～6 表示。

### 3．将日期转化成字符串

Date 对象提供一些特有的方法将日期转换为字符串，而不需要开发人员编写专门的函数去实现该功能，如表 16-3 所示。

**表 16-3　日期转换成字符串的方法及说明**

| 方　法　名 | 说　　明 |
|---|---|
| toString() | 把 Date 对象转换为字符串 |
| toLocaleString() | 根据本地时间格式，把 Date 对象转换为字符串 |
| toLocaleTimeString() | 根据本地时间格式，把 Date 对象的时间部分转换为字符串 |
| toLocaleDateString() | 根据本地时间格式，把 Date 对象的日期部分转换为字符串 |

**【例 16-1-3】**日期转换成字符串的应用。代码如下所示，页面效果如图 16-3 所示。

视频讲解

```
<!-- edu_16_1_3.html -->
<!doctype html>
<html lang="en">
    <head>
        <meta charset="UTF-8">
        <title>日期转换成字符串的应用</title>
    </head>
    <body>
        <h4>日期转化为字符串的应用 </h4>
        <script type="text/javascript">
            var MyDate=new Date();
            var msg="";
            msg+="当前日期字符串toString(): "+MyDate.toString()+"<br>";
            msg+="本地日期字符串toLocaleString(): "+MyDate.
             toLocaleString()+"<br>";
            document.write(msg);
        </script>
```

```
17     </body>
18 </html>
```

**代码解释**

代码中第 11 行定义了一个日期对象 MyDate，代表了当前的日期时间。第 13 行、第 14 行分别调用日期对象转换成字符串的相关方法将 MyDate 转换成字符串，并将结果显示在页面上。




图 16-3　日期转换成字符串实例

### 16.1.3　Math

Math 对象拥有一系列的属性和方法，能够进行比基本算术运算更为复杂的运算。但 Math 对象所有的属性和方法都是静态的，并不能生成对象的实例，但能直接访问它的属性和方法。

**1．使用 Math 的属性**

Math 的属性及说明如表 16-4 所示。

例如，计算一个圆的面积时，圆周率就可以用 Math.PI 来代替了。

```
var radius = 18;
var area = Math.PI*radius*radius;
```

**表 16-4　Math 属性及说明**

| 属　性　名 | 说　　明 |
| --- | --- |
| Math.E | 返回算术常量 e，即自然对数的底数（约等于 2.718） |
| Math.LN2 | 返回 2 的自然对数（约等于 0.693） |
| Math.LN10 | 返回 10 的自然对数（约等于 2.302） |
| Math.LOG2E | 返回以 2 为底的 e 的对数（约等于 1.414） |
| Math.LOG10E | 返回以 10 为底的 e 的对数（约等于 0.434） |
| Math.PI | 返回圆周率（约等于 3.14159） |
| Math.SQRT1_2 | 返回 2 的平方根的倒数（约等于 0.707） |
| Math.SQRT2 | 返回 2 的平方根（约等于 1.414） |

**2．使用 Math 的方法**

Math 的方法及说明如表 16-5 所示。

**表 16-5　Math 方法及说明**

| 方　法　名 | 说　　明 |
| --- | --- |
| Math.ceil(x) | 对数进行上舍入。返回大于等于 x，并且与 x 接近的整数 |
| Math.floor(x) | 对数进行下舍入。返回小于等于 x，并且与 x 接近的整数 |

续表

| 方　法　名 | 说　　明 |
|---|---|
| Math.round(x) | 把数四舍五入为最接近的整数 |
| Math.random() | 返回 0～1 之间的随机数 |
| Math.max(x,y) | 返回 x 和 y 中的最大值 |
| Math.min(x,y) | 返回 x 和 y 中的最小值 |
| Math.sqrt(x) | 返回数的平方根 |
| Math.exp(x) | 返回 e 的指数 |
| Math.pow(x,y) | 返回 x 的 y 次幂 |
| Math.log(x) | 返回数的自然对数（底为 e） |

Math 对象提供很多的数学方法用于基本运算，这些基本运算能够满足 Web 应用程序的要求。例如，在 JavaScript 脚本中，可使用 Math 对象的 random()方法生成 0～1 的随机数。

**【例 16-1-4】**使用 Math 对象产生任意范围的 10 个随机整数。代码如下所示，页面效果如图 16-4 所示。

视频讲解




图 16-4　随机数发生器实例

```
<!--  edu_16_1_4.html -->
<!doctype html>
<html lang="en">
    <head>
        <meta charset="UTF-8">
        <title>随机产生[m,n]区间内10个整数</title>
        <script  type="text/javascript">
            function $(id){return document.getElementById(id);}//获取元素
            function createInt()
            {
                var m=parseFloat($("minN").value); //解析为实数
                var n=parseFloat($("maxN").value); //解析为实数
                var array_int=new Array();
                if(m>=n)                              //合法性检验
                {  alert("数组下限值不能大于或等于上限值! 请重新输入");
                    $("minN").focus();                //让文本框自动获取焦点
                }else {
                    for(var i=0;i<10;i++)
                    {                                 //产生 m-n 之间的随机数
                        array_int[i]=Math.round((Math.random()*(n-m)+m));
                    }
                }
                $("array_num").value=array_int.join(","); //会写入文本框内
            }
```

```
25        </script>
26    </head>
27    <body>
28        <h3>随机产生[m,n]区间内10个整数</h3>
29        <form name="Form1">
30            下限: <input type="text" name="minN" id="minN" size="20"
              value=10>
31            上限: <input type="text" name="maxN" id="maxN" size="20"
              value=90><br><br>
32            产生数组: <input type="text" name="" id="array_num" size="40"
              readonly><br><br>
33            <input type="button" value="产生10个随机整数" onclick=
             "createInt();">
34            <input type="reset">
35        </form>
36    </body>
37 </html>
```

上述代码中第 7～25 行定义了两个函数，分别为$(id)、createInt()；第 20 行利用循环给数组元素赋值，随机产生[m,n]之间的整数；第 29～35 行定义了一个表单，该表单包含三个文本输入框和一个按钮并为按钮设置了 onClick 事件句柄。当用户在文本输入框中输入随机数的上限与下限后，单击“产生 10 个随机整数”按钮时会触发 Click 事件，调用 createInt()函数产生 10 个符合条件的随机整数，并在第 3 个文本框中输出。

产生[m，n]区域内随机整数的方法为：

```
var randomInt= Math.round((Math.random()*(n-m)+m));
```

## 16.1.4　Number

使用强制类型转换函数 Number(value)可以把给定的值转换成数字（可以是整数或浮点数）。Number()的强制类型转换与 parseInt()和 parseFloat()方法的处理方式相似，只是它转换的是整个值，而不是部分值。

```
var ss=Number(false) ;          //返回值为0
var ss=Number(true) ;           //返回值为1
var ss=Number(null) ;           //返回值为0
var ss=Number(100) ;            //返回值为100
var ss=Number("5.5 ");          //返回值为5.5
var ss=Number("56 ")  ;         //返回值为56
var ss=Number(undefined) ;      //返回值为NaN
var ss=Number("5.6.7 ")  ;      //返回值为NaN，与parseFloat("5.6.7")不同
var ss=Number(new Object()) ;   //返回值为NaN
```

## 16.1.5　String

String 对象是与原始字符串数据类型相对应的 JavaScript 本地对象，属于 JavaScript 核心对象之一，主要提供诸多方法实现字符串检查、抽取子串、字符串连接、字符串分割等字符串相关操作，可以通过如下方式生成 String 对象。例如：

```
var s1 = "hello,world";
var s2 = new String("hello,world");
```

此外，强制类型转换 String(value)可以把给定的值转换成字符串。

```
var s1=String("100");             //返回值为字符串100
var s1=String("acdd");            //返回值为字符串acdd
var s1=String("false");           //返回值为字符串false
var s1=String(true);              //返回值为字符串true
var s1=String(null) ;             //返回值为字符串null
var s1=new Array("111","222","333");alert(String(s1));
                                  //返回值为111,222,333
var s1=String(new Object())       //返回值为字符串[object,Object]
```

**1. 获取 String 对象长度属性**

String 对象常用的属性有 length，返回目标字符串中字符数目。例如：

```
var s1 = "hello,world";
var len = s1.length;              //s1.length返回11，s1所指向的字符串有11个字符
```

**2. 连接两个字符串**

String 对象的 concat()方法能将作为参数传入的字符串加入到调用该方法的字符串的末尾，并将结果返回给新的字符串。例如：

```
var targetString=new String("Welcome to ");
var strToBeAdded=new String("the world!");
var finalString=targetString.concat(strToBeAdded);
```

**3. 把字符串分割为字符串数组**

split()方法可以把字符串分割成字符串数组。例如“How are you doing today?”的 5 个单词之间都用空格间隔，就可以把这个字符串按照空格分成 5 个字符串，代码如下所示：

```
<script type=text/javascript>
    var str1 = " How are you doing today?";
    var subarray =str1.split(" ");  //subarray是一个数组
    for(var i=0;i<subarray.length;i++)
    {
        document.write(subarray [i]);
        document.write("<br>");
    }
</script>
```

split()方法的返回值是字符串数组。可用 Array 对象的方法访问字符串数组中的元素。split()方法分割方法还有很多。例如：

```
var sub1 = str1.split("");  //把字符串按字符分割，返回数组["H","o","w",…]
var sub2 = str1.split("o"); //把字符串按字符o分割，返回数组["H","w are y",
                             "ud","ing t","day?"]
```

**4. String 对象的显示风格方法**

String 对象还提供了可以改变字符串在 Web 页面中的显示风格的方法，如表 16-6 所示。

表 16-6　字符串显示风格的方法及说明

| 方　法　名 | 说　　明 | 方　法　名 | 说　　明 |
|---|---|---|---|
| blink() | 显示闪动字符串 | big() | 使用大字号来显示字符 |
| bold() | 使用粗体显示字符串 | small() | 使用小字号来显示字符串 |
| fontcolor() | 使用指定的颜色来显示字符串 | strike() | 使用删除线来显示字符串 |
| fontsize() | 使用指定的尺寸来显示字符 | sub() | 把字符串显示为下标 |
| Italics() | 使用斜体显示字符串 | sup() | 把字符串显示为上标 |

【例 16-1-5】字符串对象的不同显示风格，代码如下所示，页面效果如图 16-5 所示。

视频讲解

```
<!--  edu_16_1_5.html -->
<!doctype html>
<html lang="en">
    <head>
        <meta charset="UTF-8">
        <title>字符串显示风格方法的应用</title>
    </head>
    <body>
        <h4>字符串显示风格方法的应用</h4>
        <script type="text/javascript">
            var MyString=new String("How Are You?");
            document.write("原始字符串："+MyString+"<br><hr>");
            document.write("big()方法："+MyString.big()+"<br>");
            document.write("small()方法："+MyString.small()+"<br>");
            document.write("bold()方法："+MyString.bold()+"<br>");
            document.write("fontcolor('ff0000')方法："+MyString.fontcolor('ff0000')+ "<br>");
            document.write("fontsize(5)方法："+MyString.fontsize(5)+"<br>");
            document.write("italics()方法："+MyString.italics()+"<br>");
            document.write("strike()方法："+MyString.strike()+"<br>");
            document.write("sub()方法："+MyString.sub()+"<br>");
            document.write("sup()方法："+MyString.sup()+"<br>");
        </script>
    </body>
</html>
```




图 16-5　字符串显示风格实例

**代码解释**

代码中第 12～21 行调用字符串显示风格转换函数对字符串"How Are You?"进行各种风格的转换处理。

**5．字符串的大小写转换**

字符串对象提供了字符串中的字符大小写互相转换的方法，如表 16-7 所示。

**表 16-7　字符串大小写转换的方法及说明**

| 方　法　名 | 说　　明 | 方　法　名 | 说　　明 |
|---|---|---|---|
| toLowerCase() | 把字符串转换为小写 | toUpperCase() | 把字符串转换为大写 |

## 16.1.6　Boolean

Boolean 对象是对应于原始逻辑数据类型的本地对象，它具有原始的 Boolean 值，只有 true 和 false 两个状态。在 JavaScript 脚本中，1 代表 true 状态，0 表 false 状态。创建 Boolean 对象时可以用如下的语句：

```
var boolean1 = new Boolean(value); //构造方法
var boolean2 = Boolean(value);     //转换函数
```

第 1 句通过 Boolean 对象的构造函数创建对象的实例 boolean1，并用以参数形式传入的 value 值将其初始化；第 2 句使用 Boolean( )函数创建 Boolean 对象的实例 boolean2，并用以参数形式传入的 value 值将其初始化。

```
var b1 = Boolean("");              //空字符串转换为false
var b2 = Boolean("hello");         //非空字符串转换为true
var b1 = Boolean(50);              //非零数字转换为true
var b1 = Boolean(null);            //null转换为false
var b1 = Boolean(0);               //零转换为false
var b1 = Boolean(new object());    //对象转换为true
```

需要注意的是，如果省略 value 参数，或者设置为 0、-0、null、""、false、undefined 或 NaN，则该对象设置为 false；否则设置为 true（即使 value 参数是字符串"false"）。

下面所有的代码行均会创建初始值为 false 的 Boolean 对象：

```
var myBoolean=new Boolean();
var myBoolean=new Boolean(0);
var myBoolean=new Boolean(null);
var myBoolean=new Boolean("");
var myBoolean=new Boolean(false);
var myBoolean=new Boolean(NaN);
```

下面所有的代码行均会创建初始值为 true 的 Boolean 对象：

```
var myBoolean=new Boolean(1);
var myBoolean=new Boolean(true);
var myBoolean=new Boolean("true");
var myBoolean=new Boolean("false");
var myBoolean=new Boolean("Bill Gates");
```

Boolean 对象主要有 3 个方法，分别是 toSource( )、toString( )及 valueOf( )方法。toSource

方法返回表示当前 Boolean 对象实例创建代码的字符串；toString 方法返回当前 Boolean 对象实例的字符串（"true"或"false"）；valueOf 方法得到一个 Boolean 对象实例的原始 Boolean 值。

# 16.2 HTML DOM

## 16.2.1 DOM 简介

document 对象是客户端 JavaScript 最为常用的对象之一，在浏览器对象模型中，它位于 window 对象的下一层级。document 对象包含一些简单的属性，提供了有关浏览器中显示文档的相关信息，例如，该文档的 URL、字体颜色，修改日期等。另外，document 对象还包含一些引用数组的属性，这些属性可以代表文档中的表单、图像、链接、锚以及 applet。同其他对象一样，document 对象还定义了一系列的方法，通过这些方法，可以使 JavaScript 在解析文档时动态地将 HTML 文本添加到文档中。

正是由于 document 对象特有的重要性，所以从它出现开始，就在不停地扩展。遗憾的是，一开始 document 对象的扩展并没有统一的规范，不同的浏览器有不同的定义，而且彼此不兼容。为了解决不兼容带来的问题，万维网联盟（W3C）制定了一种规范，目的是创建一个通用的文档对象模型（Document Object Model，DOM），得到所有浏览器的支持。DOM 也是一个发展中的标准，它指定了 JavaScript 等脚本语言访问和操作 HTML 或者 XML 文档各个结构的方法，随着技术的发展和需求的变化，DOM 中的对象、属性和方法也在不断地变化。

DOM 的设计是以对象管理组织（OMG）的规约为基础的，因此可以用于任何编程语言。最初人们认为它是一种让 JavaScript 在浏览器间进行移植的方法，不过 DOM 的应用已经远远超出这个范围。DOM 技术使得用户页面可以动态地变化，如可以动态地显示或隐藏一个元素、改变元素的属性、增加一个元素等，DOM 技术使得页面的交互性大大增强。

## 16.2.2 DOM 节点树

HTML DOM 定义了访问和操作 HTML 文档的标准方法。DOM 将 HTML 文档表达为树结构，如图 16-6 所示。HTML 文档结构好像倒置的一棵树，其中<html>标记就是树的根节点，<head>、<body>是树的两个子节点。这种描述页面标记关系的树形结构称为 DOM 节点树（文档树）。

**【例 16-2-1】**编写如图 16-6 所示的 DOM 节点树对应的 HTML 文档。


视频讲解

```
<!--  edu_16_2_1.html -->
<!doctype html>
<html lang="en">
    <head>
        <meta charset="UTF-8">
        <title>DOM节点树的应用</title>
    </head>
    <body>
        <h1>欢迎您回到祖国怀抱</h1>
        <a href="http://www.gov.cn/">中央人民政府</a>
```

```
11    </body>
12 </html>
```

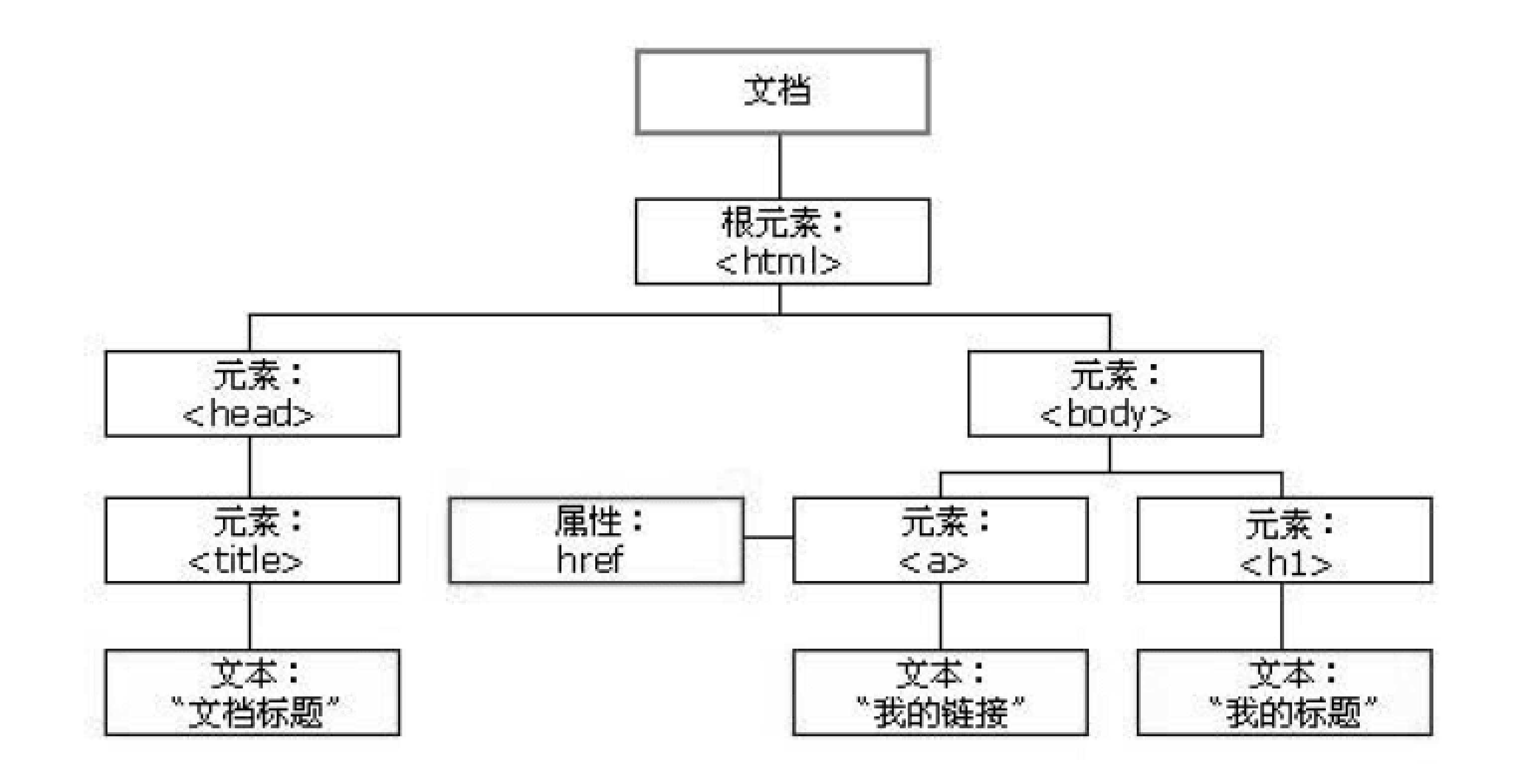



图 16-6　DOM 节点树

例 16-2-1 中的页面由<html>、<head>、<title>、<body>、<h1>、<a>等标记组成。图 16-6 展示的 DOM 节点树模型就是对【例 16-2-1】中所包含的文档结构的说明。

## 16.2.3　DOM 节点

根据 HTML DOM 规范，HTML 文档中的每个成分都是一个节点。具体的规定如下：

- 整个文档是一个文档节点。
- 每个 HTML 标记是一个元素节点。
- 包含在 HTML 元素中的文本是文本节点。
- 每一个 HTML 属性是一个属性节点。
- 注释属于注释节点。

通过 document 对象的 documentElement 属性可以获得整个 DOM 节点树上的任何一个元素。例如：

```
var root=document.documentElement;           //获取根节点
```

通过节点的 firstChild 和 lastChild 属性来获得它的第一个和最后一个子节点。DOM 规定一个页面只有一个根节点，根节点是没有父节点的，除此之外，其他节点都可以通过 parentNode 属性获得自己的父节点，例如：

```
document.write(root.firstChild.nodeName);   //输出HEAD
document.write(root.lastChild.nodeName);    //输出BODY
var parentNode=bNode.parentNode;            //parentNode属性
```

同一父节点下位于同一层次的节点称为“兄弟节点”，一个子节点的前一个节点可以用 previousSibling 属性获取，对应的后一个节点可以用 nextSibling 属性获取。在图 16-6 中，head 节点下的子节点 title 节点以及 script 节点就互为“兄弟节点”。 从 DOM 树中可以看出根节点没有父节点，而最末端的节点没有子节点。不同节点对应的 HTML 元素是不同的，因此节点有不同类型。文档树中每个节点对象都有 nodeType 属性，该属性返回节点的类型，

常用的节点类型及其说明如表 16-8 所示。

```
var nodeList=root.childNodes;
document.write(nodeList[0].nextSibling.nodeName) ;     //输出BODY
document.write(nodeList[1].previousSibling.nodeName);  //输出HEAD
```

**表 16-8　常用节点类型及说明**

| 节 点 类 型 | nodeType 值 | 说　　明 |
|---|---|---|
| Element | 1 | 元素节点，表示文档中的 HTML 元素 |
| Attr | 2 | 属性节点，表示文档中 HTML 元素的属性 |
| Text | 3 | 文本节点，表示文档中的文本内容 |
| Comment | 8 | 注释节点，表示文档中的注释内容 |
| Document | 9 | 文档节点，表示当前文档 |

从表 16-8 中可以看出，如果某个节点的 nodeType 的值为 9，则说明该节点对象为一个 Document 对象，如果某个节点的 nodeType 值为 1，则说明节点对象为一个 Element 对象。不同类型的节点还可以包含其他类型的节点，相互连接在一起就构成了一个完整的树形结构。对于大多数 HTML 文档来说，元素节点、文本节点及属性节点是必不可少的。

**1．元素节点（Element Node）**

元素节点构成了 DOM 基础。在文档结构中，<html>、<head>、<body>、<h1>、<p>和<ul>等标记都是元素节点。各种标记提供了元素的名称，如文本段落元素的名称是 p，无序列表元素的名称是ul 等。元素可以包含其他元素，也可以被其他元素包含。图 16-6 显示了这种包含与被包含的关系，唯独 html 元素没有被其他元素包含，因为它是根元素，代表整个文档。

**2．文本节点（Text Node）**

元素节点只是节点树中的一种类型，如果文档完全由元素组成，那么这份文档本身将不包含任何信息，因此文档结构也就失去了存在的价值。在 HTML 文档中，文本节点包含在元素节点内，如 h1、p、li 等节点就可以包含一些文本节点。

**3．属性节点（Attribute Node）**

元素一般都会包含一些属性，属性的作用是对元素做出更具体的描述。例如，一般元素都有 title 属性，该属性能够对元素进行详细地描述或说明，以便用户了解该元素的用途、作用或功能。示例如下：

```
<img src="image2.jpg" title="三星手机"  />
```

在上例的 img 标记中，title 就是一个属性节点，由于属性总是被放在起始标记内，所以属性节点总是被包含在元素节点当中，可以通过元素节点对象调用 getAttribute( )方法来获取属性节点。

## 16.2.4　DOM 节点访问

访问节点的方式有很多种，可以通过 document 对象的方法来访问节点，也可以通过元素节点的属性来访问节点。结合【例 16-2-2】来进行具体分析。

【例 16-2-2】 DOM 节点访问的应用。代码如下所示。

视频讲解

```
<!--  edu_16_2_2.html -->
<!doctype html>
<html lang="en">
    <head>
        <meta charset="UTF-8">
        <title>DOM节点访问</title>
        <script type="text/javascript">
            function validate()
            {
                 //此处为用户登录时的校验处理代码
            }
        </script>
    </head>
    <body>
        <form method="post" action="" name="myform">
            <fieldset style="width:350px;height:150px;text-align:center;">
                <legend align="center">用户信息</legend>
                用户名:<input type="text" name="username" id="username"><br>
                密码:<input type="password" name="password" id="password"><br>
                邮箱:<input type="text" name="email" id="email"><br>
                <input type="button" value="提交" onclick="validate();">
                <input type="reset">
            </fieldset>
        </form>
    </body>
</html>
```

如果要对【例 16-2-2】中的用户名文本输入框、密码输入框及邮箱地址文本输入框进行访问的话，可以通过如下几种方式进行访问。

**1．通过 getElementById( )方法访问节点**

document 对象的 getElementById()方法可以访问页面中的节点，该方法在使用时，必须指定一个目标元素的 id 作为参数。

1）基本语法

```
var s=document.getElementById(id);  //调用时参数需要加双引号
```

在使用该方法时需要注意以下两点：

- id 为必选项，对应于页面元素属性 id 的属性值，类型为字符串型。在页面设计时最好给每一个需要交互的元素设定一个唯一的 id，以便查找。
- 该方法返回的是一个页面元素的引用，如果页面上出现了不同元素使用了同一个 id，则该方法返回的只是第一个找到的页面元素；如果给定的 id 没有找到对应的元素，则返回 null。

通过此方法可以编写一个通过 id 获取 HTML 文档上元素的通用方法$(id)。

```
function $(id){return document.getElementById(id);} //调用时参数需要加双引号
```

对【例 16-2-2】中的脚本做一些修改，当用户输入用户名、密码及邮箱地址后，单击提交按钮，触发该按钮的单击事件，调用其绑定的事件处理函数 validate()，通过告警消息框显示用户输入的用户名、密码及邮箱等信息。代码如下所示：

```
1 <script type="text/javascript">
2     function $(id){return document.getElementById(id);}
3     function validate(){
4         var msg ="用户名为: "
5         var username = $("username").value;
6         var psw = $("password").value;
7         var email = $("email").value;
8         msg=msg+username+"\n密码为:"+ psw+"\n邮箱地址为:"+ email;
9         alert(msg);     //输出
10    }
11 </script>
```

2）代码解释

代码中用户名文本输入框、密码输入框、邮箱文本输入框的 id 分别为 username、password、email。代码中定义了两个函数，分别是$(id)和 validate()，其中，$(id)功能是通过 id 获取 HTML 页面上的任一元素；validate()功能是通过$(id)函数获取特定元素，并获取该元素的 value 值。代码中第 5～7 行通过 id 获取每个文本框中输入的值，然后通过告警消息框输出信息。

**2．通过 getElementsByName( )方法访问节点**



视频讲解

除通过元素的 id 可以获取对象外，还可以通过元素的名字来访问。

1）基本语法

```
var s=document.getElementsByName("name");
```

在使用该方法时需要注意以下两点：

- name 为必选项，对应于页面元素属性 name 的属性值，类型为字符串型。该方法调用时返回的是一个数组，即使对应于该名字的元素只有一个。
- 如果指定名字，在页面中没有相应的元素存在，则返回一个长度为 0 的数组，程序中可以通过判断数组的 length 属性值是否为 0 判断是否找到了对应的元素。

通过此方法可以编写一个通过 name 获取 HTML 文档上的一组元素的通用方法$name(name)，此函数返回一个对象数组。

```
function $name(name){return document.getElementsByName(name);}
//调用时参数需要加双引号
```

如果将 JavaScript 程序中的 getElementById()方法替换成 getElementsByName()方法来获取用户名、密码及邮箱地址，则脚本代码需要做如下修改：

```
<script type="text/javascript">
    function $name(name){return document.getElementsByName(name);}
    function validate(){
        var msg ="用户名为: "
        var username=$name("username")[0].value;  //取用户名
        var psw=$name("password")[0].value;       //取密码
        var email=$name("email")[0].value;        //取邮箱
        msg=msg+username+"\n密码为:"+ psw+"\n邮箱地址为:"+ email;
        alert(msg);                               //输出
    }
</script>
```

2）代码解释

代码中用户名文本输入框、密码输入框、邮箱文本输入框的 name 分别为 username、password、email。代码中定义了两个函数，分别是$name(name)和 validate()，其中，$name(name)功能是通过 name 获取 HTML 页面上的特定元素数组；validate()功能是通过$name(name)函数获取特定元素数组中第 0 个元素，格式为$name("username")[0]，并获取该元素的 value 值，代码中第 5～7 行可以获取每个文本框中输入的内容，然后通过告警消息框输出信息。

**3．通过 getElementsByTagsName( )方法访问节点**

视频讲解

除了通过元素的 id 和 name 可以获得对应的元素外，还可以通过标记名称来获得页面上所有同类的元素，如表单中的所有 input 元素。

1）基本语法

```
var s=document.getElementsByTagName(tagname);
```

在使用该方法时需要注意以下两点：

- tagname 为必选项，对应于页面元素的类型，为字符串型的数据。该方法调用时返回的是一个数组，即使页面中对应于该类型的元素只有一个。
- 通过判断数组的 length 属性值来获知页面上该类型元素的总数。

通过此方法可以编写一个通过 tagname 获取 HTML 文档中的一组元素的通用方法$tag(tagname)，此函数返回一个对象数组。

```
function $tag(tagname){return document.getElementsByTagName(tagname);}
//调用时参数需要加双引号
```

如果在 JavaScript 程序中用 getElementsByTagName 方法来获取用户名、密码及邮箱地址，则脚本代码需要做如下修改：

```
 1 <script type="text/javascript">
 2    function $tag(tagname){return document.getElementsByTagName
      (tagname);}
 3    function validate(){
 4       var msg ="用户名为: "
 5       var username=$tag("input")[0].value;       //取用户名
 6       var psw=$tag("input")[1].value;            //取密码
 7       var email=$tag("input")[2].value;          //取邮箱
 8       msg=msg+username+"\n密码为:"+ psw+"\n邮箱地址为:"+ email;
 9       alert(msg);                                //输出
10    }
11 </script>
```

2）代码解释

在【例 16-2-2】中，由于用户名输入框、密码输入框、邮箱输入框及按钮，它们都是<input>类型的元素，所以可以一次通过$tag(tagname)函数获取页面上所有的 input 标记元素，得到的是一个 input 类型的元素数组，然后依次访问数组中的每个成员。代码中第 5～7 行通过数组的下标依次获取每个文本框的值，然后通过告警消息框输出信息。

**4．通过 form 元素访问节点**

如果要获得页面中的 form 对象，除了 getElementById( )、getElementsByName( )方法

外，还可以通过 document 对象的 forms 属性来获得这个 form 对象。表单是用户与网页进行交互的重要手段，通过表单可以一次性获取表单中大量元素的信息。获得【例 16-2-2】文档中的 form 对象的方法如下所示：

```
var myfrm = document.forms;              //通过document的forms属性获得数组对象
var myloginform = myfrm[0];              //获得数组中的第一个form对象
```

当然也可以通过 form 对象的 name 属性来访问到页面中的 form 对象。格式如下所示：

```
var myform = document.loginform;         //loginform为form对象的名称
```

获得 form 对象之后，如果想得到 form 对象包含的其他元素，就可以通过 form 对象的 elements 属性或该元素的 name 属性来获得，例如，前面代码获得了 form 对象，可以通过如下程序获得该 form 对象包含的用户输入框、密码框或邮箱地址框。

```
var username1=loginform.elements[0];   //通过elements属性来访问用户名输入框
var username2=loginform.username;      //通过name属性来访问用户名输入框
var password1=loginform.elements[1];   //通过elements属性来访问密码输入框
var password2=loginform.password;      //通过name属性来访问密码输入框
var email1=loginform.elements[2];      //通过elements属性来访问邮箱地址输入框
var email2=loginform.email;            //通过name属性来访问邮箱地址输入框
```

## 16.2.5 DOM 节点操作

前面已经学过了如何访问文档中的不同节点，不过这仅仅是使用 DOM 所能实现的功能中的一小部分。DOM 的应用非常广泛，如可以通过 document 对象实现表格的动态添加和删除，可以通过 document 对象替换文本节点的内容等。

**1. 创建和修改节点**

document 对象有很多创建和修改不同类型节点的方法，常用方法如表 16-9 所示。

**表 16-9 创建和修改节点的方法及说明**

| 方 法 名 | 说 明 |
|---|---|
| createElement(tagname) | 创建标记名为 tagname 的节点 |
| createTextNode(text) | 创建包含文本 text 的文本节点 |
| createDocumentFragment() | 创建文档碎片 |
| createAttribute() | 创建属性节点 |
| createComment(text) | 创建注释节点 |
| removeChild(node) | 删除一个名为 node 的子节点 |
| appendChild(node) | 添加一个名为 node 的子节点 |
| insertBefore(nodeB,nodeA) | 在名为 nodeA 节点前插入一个名为 nodeB 的节点 |
| replaceChild(nodeB,nodeA) | 用一个名为 nodeB 节点替换另一个名为 nodeA 的节点 |
| cloneNode(boolean) | 克隆一个节点，它接收一个 boolean 参数，为 true 时表示该节点带文字；为 false 时表示该节点不带文字 |

假设要在一个 HTML 页面中添加一个<p>节点，<p>节点内的文本内容是“Hello World!”，在此可以使用 createElement()、createTextNode()及 appendChild()方法来实现。

【例 16-2-3】运用 document 对象在网页中创建文本节点。其页面效果如图 16-7 所示。

视频讲解

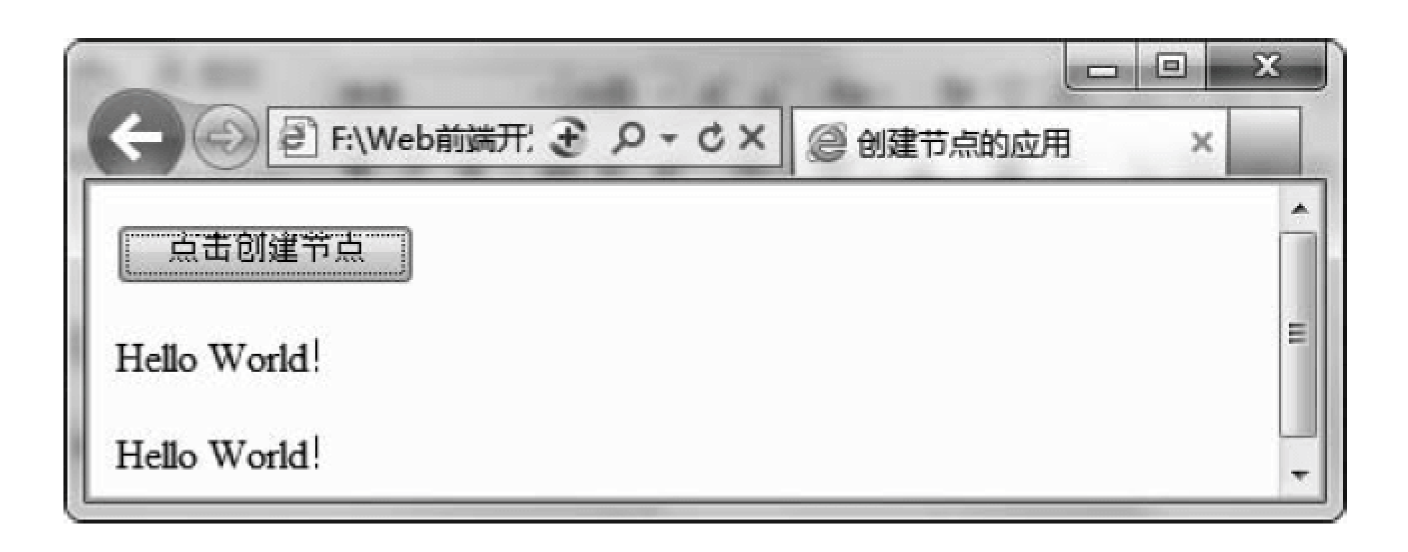



图 16-7　创建节点实例

在此例中创建段落 p 元素并为段落设置文本节点内容，共分四个步骤：

```
var newp = document.createElement("p"); //第1步，创建p元素节点
var ptext = document.createTextNode("hello world! ");//第2步，创建文本节点
newp.appendChild(ptext);                //第3步，将文本节点加入到p元素中
document.forms[0].appendChild(newp);    //第4步，将元素p节点插入到表单form中
```

按照以上给定的步骤编写代码如下所示：

```
 1 <!--  edu_16_2_3.html -->
 2 <!doctype html>
 3 <html lang="en">
 4     <head>
 5         <meta charset="UTF-8">
 6         <title>创建节点的应用</title>
 7         <script  type="text/javascript">
 8             function createP(){
 9                 var newp =document.createElement("p");
10                 var ptext = document.createTextNode("Hello World! ");
11                 newp.appendChild(ptext);
12                 document.forms[0].appendChild(newp);
13             }
14         </script>
15     </head>
16     <body>
17         <form name="form1">
18             <input type="button" value="点击创建节点" onClick="createP()">
19         </form>
20     </body>
21 </html>
```

上述代码中第 8～13 行定义了一个 JavaScript 函数，名为 createP()；第 17～19 行定义了一个表单，在表单中插入一个普通按钮，并为该按钮的 onClick 事件句柄绑定了事件处理函数 createP()。当单击按钮时会触发 Click 事件调用 createP()向表单中添加节点<p>，并将其文本内容设置为“Hello World!”。

除了添加一个节点外，也可以使用 removeChild()、insertBefore()及 replaceChild()方法删除、插入和替换节点。

视频讲解

【例 16-2-4】节点删除、插入和替换。其页面效果如图 16-8 所示。



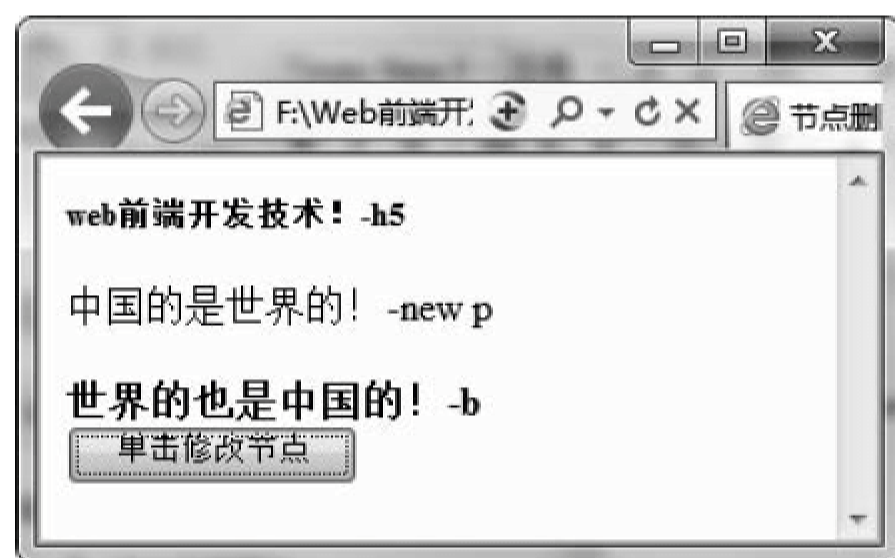



图 16-8　节点修改前/后界面

```
1 <!--  edu_16_2_4.html -->
2 <!doctype html>
3 <html lang="en">
4     <head>
5         <meta charset="UTF-8">
6         <title>节点删除、插入、替换</title>
7         <script type="text/javascript">
8             function $tag(tagname){
                  return document.getElementsByTagName(tagname);}
9             function operateNode(){
10                //删除<p>元素
11                var p =$tag("p")[0];
12                document.form1.removeChild(p);
13                //将<h2>元素更换为<h5>元素，并重新设置文本节点内容
14                var h5 =document.createElement("h5");
15                var ptext = document.createTextNode("web前端开发技术！-h5");
16                h5.appendChild(ptext);
17                var h2 = $tag("h2")[0];
18                document.form1.replaceChild(h5,h2);
19                //b元素前插入一个<p>元素
20                var newp =document.createElement("p");
21                var ptext1 = document.createTextNode("中国的是世界的！
                 -new p");
22                newp.appendChild(ptext1);
23                document.form1.insertBefore(newp,$tag("b")[0]);
24            }
25        </script>
26    </head>
27    <body>
28        <form name="form1">
29            <h2>javaScript程序设计-h2</h2>
30            <p>hello world!-p</p>
31            <b>世界的也是中国的！-b</b><br>
32            <input type="button" value="单击修改节点" onClick=
             "operateNode()">
33        </form>
34    </body>
35 </html>
```

上述代码中第 7～25 行定义了两个 JavaScript 函数，分别名为$tag(tagname)、operateNode()；第 28～33 行定义了一个表单对象，表单中有<h2>、<p>、<b>、<input>四个节点，并设置普通按钮的 onClick 属性值为 operateNode()。当单击按钮时会触发 Click 事件调用 operateNode()，执行其中的代码。在代码执行过程中，首先会将页面上的<p>节点

删除，然后将<h2>节点替换成<h5>节点并将<h5>节点中的文本内容设置为“web 前端开发技术！-h5”，最后在<b>节点前插入一个<p>节点并将该节点中的文本内容设置为“中国的是世界的！-new p”。

除了以上例子中介绍的方法用来创建和修改节点之外，还可以使用 cloneNode()方法复制一个节点，使用 createDocumentFragment()方法创建文档片段，在此就不一一举例了。

**2．节点的 innerText 和 innerHTML 属性**

在 DOM 中有两个很重要的属性，分别是 innerText 和 innerHTML，通过这两个属性，可以更方便地进行文档操作。

innerText 属性是用来修改起始标记和结束标记之间的文本。例如，假设有个空的<div>节点，如果希望在该<div>设置文本内容为“中国你好！!”，则按照前面的介绍，代码需要这样编写：

```
oDiv.appendChild(document.createTextNode("中国你好!! ");
```

如果使用 innerText，代码就可以这样编写：

```
oDiv.innerText="中国你好!! ";
```

使用 innerText，代码更加简洁，且更容易理解。另外，innerText 会自动将小于号、大于号、引号和&符号进行 HTML 编码，所以不需要担心这些特殊字符。

innerHTML 属性可以直接给元素分配 HTML 字符串，而不需考虑使用 DOM 的方法来创建元素。例如，为空的<div>节点创建<strong>子节点，运用 DOM 方法创建的代码如下：

```
var strong1 =document.createElement("strong");
var otext = document.createTextNode("hello world! ");
strong1.appendChild(otext);
oDiv.appendChild(strong1);
```

如果使用 innerHTML 属性，代码变成：

```
oDiv.innerHTML="<strong> hello world! </strong>";
```

使用 innerHTML 属性，四行代码变成一行，通俗易懂！

还可以使用 innerText 和 innerHTML 属性获取元素的内容。如果元素只包含文本，则 innerText 和 innerHTML 返回相同的值。但是，如果同时包含文本和其他元素，innerText 将只返回文本的内容，而 innerHTML 将返回所有元素和文本的 HTML 代码。

**【例 16-2-5】**document 对象的 innerText 和 innerHTML 属性的应用。代码如下所示，页面效果如图 16-9 所示。

视频讲解

```
<!--  edu_16_2_5.html -->
<!doctype html>
<html lang="en">
   <head>
      <meta charset="UTF-8">
      <title>innerText、innerHTML举例</title>
      <script type="text/javascript">
         function textGet(){
            var oDiv = document.getElementById("oDiv");
```

```
10                  var msg = "通过innerText属性获得：";
11                  msg+=oDiv.innerText;
12                  msg+="\n通过innerHTML属性获得：  "
13                  msg+=oDiv.innerHTML;
14                  alert(msg);
15              }
16          </script>
17      </head>
18      <body onload="textGet()">
19          <div id="oDiv" >
20              <strong>web前端开发技术，不错！</strong>
21          </div>
22      </body>
23  </html>
```




图 16-9　innerText、innerHTML 属性的应用

上述代码中第 7～16 行定义了一个 JavaScript 函数，名为 textGet()；第 18 行给 body 标记设置 onload 事件句柄，绑定了事件处理函数 textGet()。窗口装载时，调用事件处理函数 textGet()，通过告警消息框输出信息，如图 16-9 所示。

**3．获取并设置指定元素属性**

在 DOM 中，如果需要动态地获取及设置节点属性的话，可以通过 getAttribute()方法、setAttribute()方法来处理，具体方法的使用说明如表 16-10 所示。

**表 16-10　获取和设置节点属性的方法及说明**

| 方　法　名 | 说　　明 |
|---|---|
| getAttribute(name) | 该方法用于获取元素指定属性的值。参数 name 为字符串，表示属性的名称 |
| setAttribute(name,value) | 该方法用于设置元素指定属性的值。参数 name 为字符串，表示要设置的属性的名称，参数 value 为字符串，表示属性的值 |

**【例 16-2-6】** DOM 节点属性获取和设置方法。代码如下所示，页面效果如图 16-10 所示。

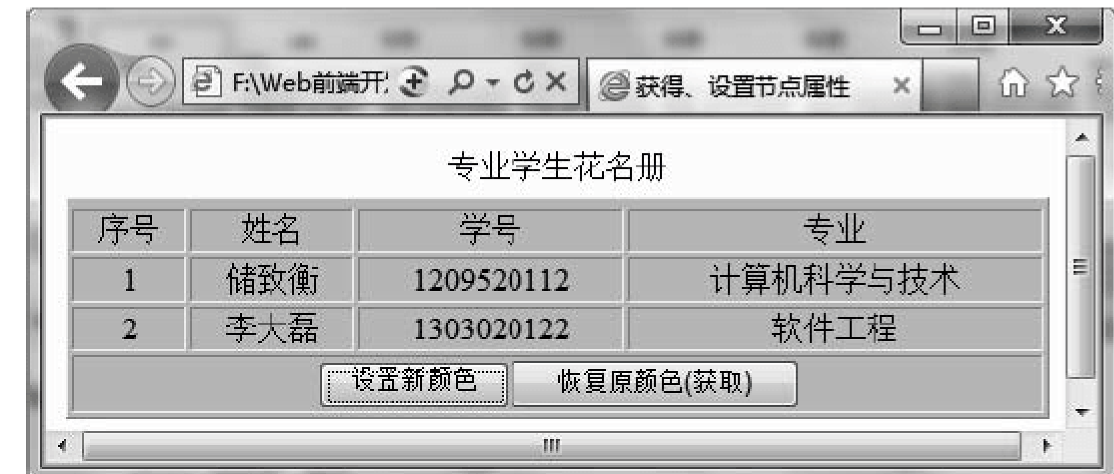
```
<!--  edu_16_2_6.html -->
<!doctype html>
<html lang="en">
    <head>
        <meta charset="UTF-8">
        <title>获得、设置节点属性</title>
        <style type="text/css">
```



视频讲解

```
            td{text-align:center;}
        </style>
        <script type="text/javascript">
            var  bg_begin;  //保存原始值
            function getBgColor(){
                bg_begin=$("myTable").getAttribute("bgColor");    //保存原始值
            }
            function $(id){return document.getElementById(id);}
            function randomInteger(){            //随机产生0-255之间的整数
                var int=Math.floor(Math.random()*256);
                return int;
            }
            function changeColor(){
                $("myTable").setAttribute("bgColor",createHexColor());
            }
            function createHexColor(){//产生十六进制颜色串#FFFFFF
                var rc=randomInteger().toString(16);   //转换为十六进制
                var gc=randomInteger().toString(16);   //转换为十六进制
                var bc=randomInteger().toString(16);   //转换为十六进制
                var color1="#"+rc+gc+bc;               //形成6位十六进制数
                return color1;
            }
            function restoreColor(){
                $("myTable").setAttribute("bgColor",bg_begin);
            }
        </script>
    </head>
    <body onload="getBgColor()">
        <form method="post" action="">
            <table id="myTable" align="center" border="1" bgColor="#99cccc" width="500px" >
                <tr>
                <caption>专业学生花名册</caption>
                <td>序号</td><td>姓名</td><td>学号</td><td>专业</td>
                </tr>
                <tr>
                <td>1</td><td>储致衡</td><td>1209520112</td>    <td>计算机科学与技术</td>
                </tr>
                <tr>
                <td>2</td><td>李大磊</td><td>1303020122</td><td>软件工程</td>
                </tr>
                <tr>
                <td colspan="4"><input type="button" value="设置新颜色" onclick="changeColor()"><input type="button" value="恢复原颜色(获取)" onclick="restoreColor()"></td>
                </tr>
            </table>
        </form>
    </body>
</html>
```






| 序号 | 姓名 | 学号 | 专业 |
|---|---|---|---|
| 1 | 储致衡 | 1209520112 | 计算机科学与技术 |
| 2 | 李大磊 | 1303020122 | 软件工程 |

图 16-10　获取、设置节点属性方法的应用

**代码解释**

代码中第 10～33 行定义了六个 JavaScript 函数，分别是$(id)、randomInteger()、changeColor()、restoreColor()、createHexColor()、getBgColor()；其中$(id)功能是通过 id 获取页面元素；randomInteger()功能是产生 0～255 的任意一个整数；changeColor()功能是改变表的背景颜色；restoreColor()功能是恢复表的上次背景颜色；createHexColor()功能是产生一个十六进制的颜色，如#FF88EE；getBgColor()功能是获取表格初始背景颜色。

第 37～51 行定义了一个表格，表格的背景颜色属性 bgColor 初始值为#99cccc。第 49 行定义了一个“设置新颜色”按钮，并为该按钮设置了 onClick 事件句柄，每单击一次调用 changeColor()更改表格的背景颜色一次。第 49 行还定义了一个“恢复原颜色(获取)”按钮，并为该按钮设置了 onClick 事件句柄，单击一次调用 restoreColor()还原为上一次表格的背景颜色。

# 16.3　BOM

在实际应用中，常常使用 JavaScript 操作浏览器窗口以及窗口上的控件，从而实现用户和页面的动态交互功能。因而浏览器预定义了很多内置对象，这些对象都含有相应的属性和方法，通过这些属性和方法控制浏览器窗口及其控件。客户端浏览器这些预定义的对象统称为浏览器对象，它们按照某种层次组织起来的模型统称为浏览器对象模型（Browser Object Model，BOM）。浏览器对象模型定义了浏览器对象的组成和相互关系，描述了浏览器对象的层次结构，是 Web 页面中内置对象的组织形式。

浏览器对象的模型如图 16-11 所示，从图中不仅可以看到浏览器对象的组成，还可以看到不同对象的层次关系，window 对象是顶层对象，包含了 document、history、location、navigator、screen 及 frame 对象。这些对象都含有若干属性和方法，使用这些属性和方法可以操作 Web 浏览器窗口中的不同对象，控制和访问 HTML 页面中的不同内容。

## 16.3.1　window 对象

window 对象位于浏览器对象模型的顶层，是 document、frame、location 等其他对象的父类。在实际应用中，只要打开浏览器，无论是否存在页面，window 对象都将被创建。由于 window 对象是所有对象的顶层对象，所以按照对象层次访问某一个对象时不必显式地注明 window 对象。

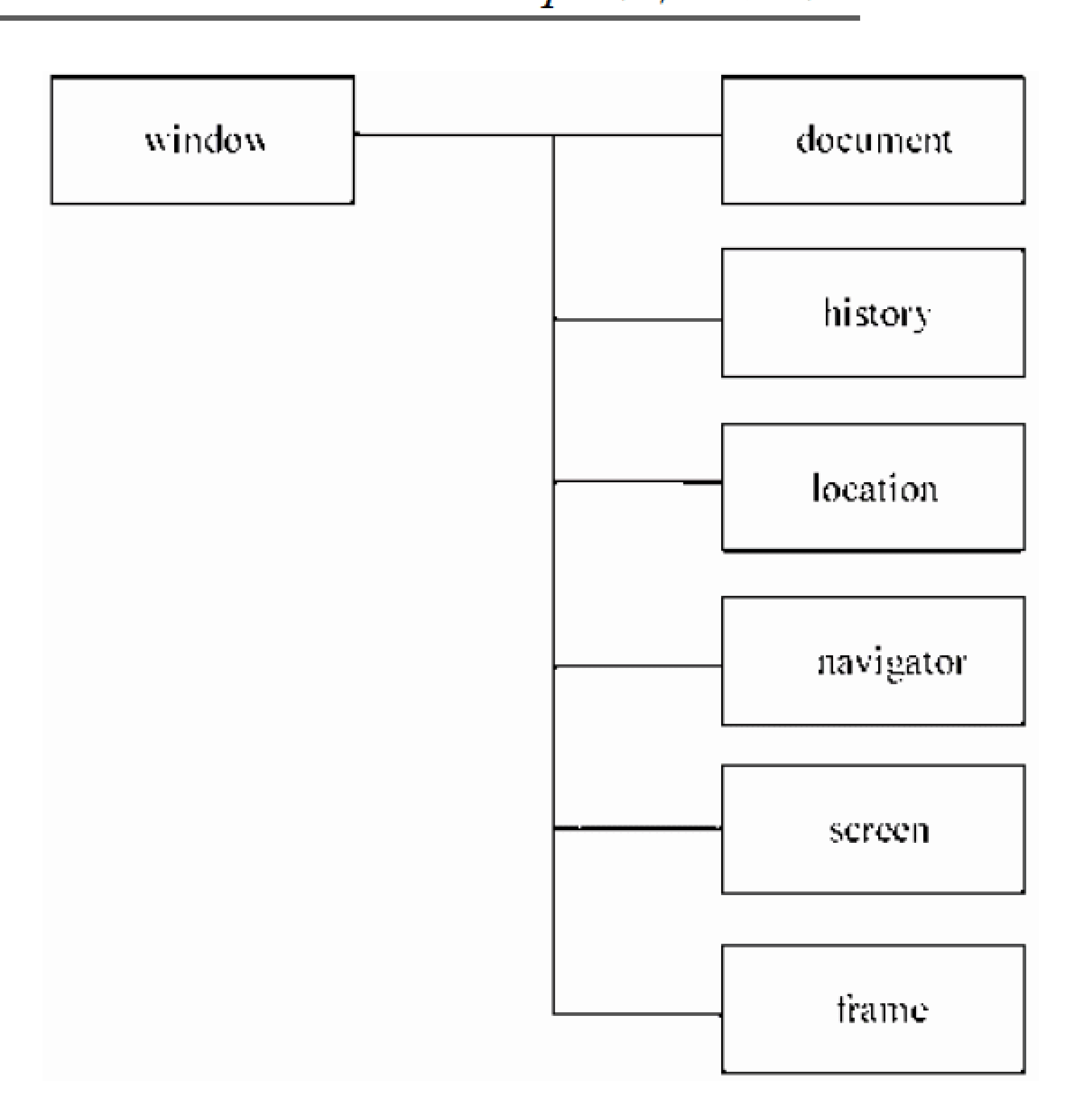



图 16-11　浏览器对象模型

window 对象内置了许多方法供用户操作，下面列出最常用的 window 对象的方法，如表 16-11 所示。

window 对象提供了三种用于客户与页面交互的对话框，分别是告警框、确认框和提示框等，这三种对话框使用方法在 14.2.3 节中已经介绍过了，在此不再重复。

window 对象还提供一些定时器方法，这些方法可以使 JavaScript 代码周期性地重复或延迟执行。例如，window 对象的 setInterval()方法用于设置在指定的时间间隔内周期性地触发某个事件，典型的应用如动态状态栏、动态显示当前时间等； clearInterval()方法用于清除该间隔定时器使目标事件的周期性触发失效。下面的例子中调用这两个方法实现窗口状态栏的移动。

**表 16-11　window 对象的方法及说明**

| 方　法　名 | 说　　明 |
|---|---|
| alert(message) | 显示带有一段消息和一个确认按钮的告警框 |
| confirm(question) | 显示带有一段消息以及确认按钮和取消按钮的对话框 |
| open(url,name,features,replace) | 打开一个新的浏览器窗口或查找一个已命名的窗口 |
| prompt(“提示信息”，默认值) | 显示可提示用户输入的对话框 |
| blur() | 把键盘焦点从顶层窗口移开 |
| close() | 关闭浏览器窗口 |
| focus() | 把键盘焦点给予一个窗口 |
| setInterval(code,interval) | 按照指定的周期（以毫秒计）来调用函数或计算表达式 |
| setTimeout(code,delay) | 在指定的毫秒数后调用函数或计算表达式 |
| clearInterval(intervalID) | 取消由 setInterval()设置的 timeout |
| clearTimeout(timeoutID) | 取消由 setTimeout()方法设置的 timeout |

**【例 16-3-1】**window 对象的定时器方法实现 div 内字符串的移动。代码如下所示，页面效果如图 16-12 所示。

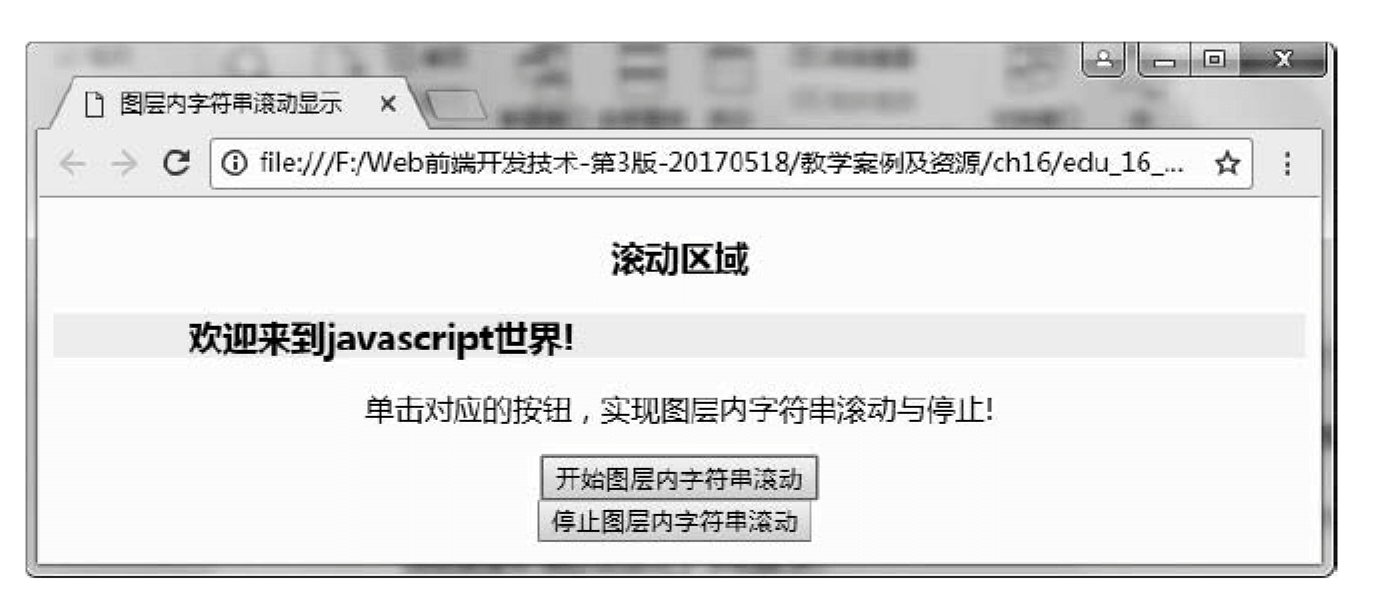



图 16-12　div 内字符串滚动界面



视频讲解

代码用下列代码全部替换。

```
<!--  edu_16_3_1.html -->
<!doctype html>
<html lang="en">
    <head>
        <meta charset="UTF-8">
        <title>图层内字符串滚动显示</title>
        <style type="text/css">
            #myDiv{width:100%;height:24px;background:#DDFFAA;}
        </style>
        <script type="text/javascript">
            var TimerID;
            var loop=1;//设置启动次数，防止多次启动
            var dir=1;//设置方向变量初值
            var str_num=0;     //用于动态显示的目标字符串
            var str="欢迎来到javascript世界!";
            function $(id){return document.getElementById(id);}
            function startMove(){
                //设定图层内动态显示的字符串信息
                var str_space="";
                str_num=str_num+1*dir;        //动态改变运动步长
                if(str_num>50 || str_num<0){    dir=-1*dir; }    //改变运动方向
                for(var i=0;i<str_num;i++){str_space+=" ";}
                $("myDiv").innerHTML="<h3>"+str_space+str+"</h3>";//动态赋值
            }
            function MyStart(){
              //图层内字符串滚动开始
                if (loop==1)   {TimerID=setInterval("startMove();",100);}
                loop++;
            }
            function MyStop(){
            //图层内字符串滚动结束，并更新图层内字符串
                clearInterval(TimerID);
                loop=1;//恢复初始值
                $("myDiv").innerHTML="<h3>图层内字符串滚动结束!</h3>";
            }
        </script>
    </head>
    <body >
        <h3 align="center">滚动区域</h3>
        <div name="" id="myDiv">欢迎来到JavaScript世界!</div>
        <div style="text-align:center;">
            <p>单击对应的按钮，实现图层内字符串滚动与停止!</p>
            <form name="MyForm">
```

```
44              <input type="button" value="开始图层内字符串滚动" onclick=
                "MyStart()"> <br>
45              <input type="button" value="停止图层内字符串滚动" onclick=
                "MyStop()"> <br>
46          </form>
47      </div>
48  </body>
49 </html>
```

**代码解释**

代码中第 10 行～第 36 行定义了 4 个 JavaScript 函数，分别为$(id)、startMove()、MyStart()、MyStop()；第 44 行、第 45 行定义了两个普通按钮，分别是“开始图层内字符串滚动”按钮及“停止图层内字符串滚动”按钮，并为这两个按钮设置了 onClick 事件句柄。

当单击“开始图层内字符串滚动”按钮时会触发事件调用 MyStar()，执行其中的代码“TimerID=setInterval("startMove();",100);”，这条语句的作用是间隔 100ms 会执行 startMove()，实现 div 内字符串的滚动效果，并把返回值赋给变量 TimerID；当单击“停止图层内字符串滚动”按钮时会触发事件调用 MyStop()，代码“clearInterval(TimerID);”的作用是清除该间隔定时器使目标事件的周期性触发失效。代码第 27 行的作用是单击 1 次“开始图层内字符串滚动”按钮时启动间隔执行 startMove()，当多次单击此按钮时不重复执行。

## 16.3.2 Navigator 对象

navigator 对象用于获取用户浏览器的相关信息。该对象是以 Netscape Navigator 命名的，在 Navigator 和 Internet Explorer 中都得到了支持。navigator 对象包含若干属性，主要用来描述浏览器的信息，但不同浏览器所支持的 navigator 对象的属性也是不同的，常用的属性如表 16-12 所示。

**表 16-12　navigator 对象的属性及说明**

| 属　性　名 | 说　　明 |
|---|---|
| appName | 返回浏览器的名称 |
| appVersion | 返回浏览器的平台和版本信息 |
| platform | 返回运行浏览器的操作系统平台 |
| systemLanguage | 返回操作系统使用的默认语言 |
| userAgent | 返回由客户机发送服务器的 user-agent 头部的值 |
| appCodeName | 返回浏览器的代码名 |

另外，navigator 对象还支持一系列的方法，与属性一样，不同浏览器支持的方法也不完全相同。常用的方法如表 16-13 所示。

**表 16-13　navigator 对象的方法及说明**

| 方　法　名 | 说　　明 |
|---|---|
| taintEnabled() | 规定浏览器是否启用数据污点(data tainting) |
| javaEnabled() | 规定浏览器是否启用 Java |
| preference() | 查询或者设置用户的优先级，该方法只能用在 Navigator 浏览器中 |
| savePreference() | 保存用户的优先级，该方法只能用在 Navigator 浏览器中 |

【例 16-3-2】navigator 对象的应用。代码如下所示，页面效果如图 16-13 所示。

视频讲解

```
1 <!--  edu_16_3_2.html -->
2 <!DOCTYPE html>
3 <html>
4     <body>
5     <div id="example"></div>
6     <script>
7         txt = "<p>1.Browser CodeName: " + navigator.appCodeName + "</p>";
8         txt+= "<p>2.Browser Name: " + navigator.appName + "</p>";
9         txt+= "<p>3.Browser Version: " + navigator.appVersion + "</p>";
10        txt+= "<p>4.Cookies Enabled: " + navigator.cookieEnabled + "</p>";
11        txt+= "<p>5.Platform: " + navigator.platform + "</p>";
12        txt+= "<p>6.User-agent header: " + navigator.userAgent + "</p>";
13        txt+= "<p>7.User-agent language: " + navigator.systemLanguage +
          "</p>";
14        document.getElementById("example").innerHTML=txt;
15    </script>
16    </body>
17 </html>
```

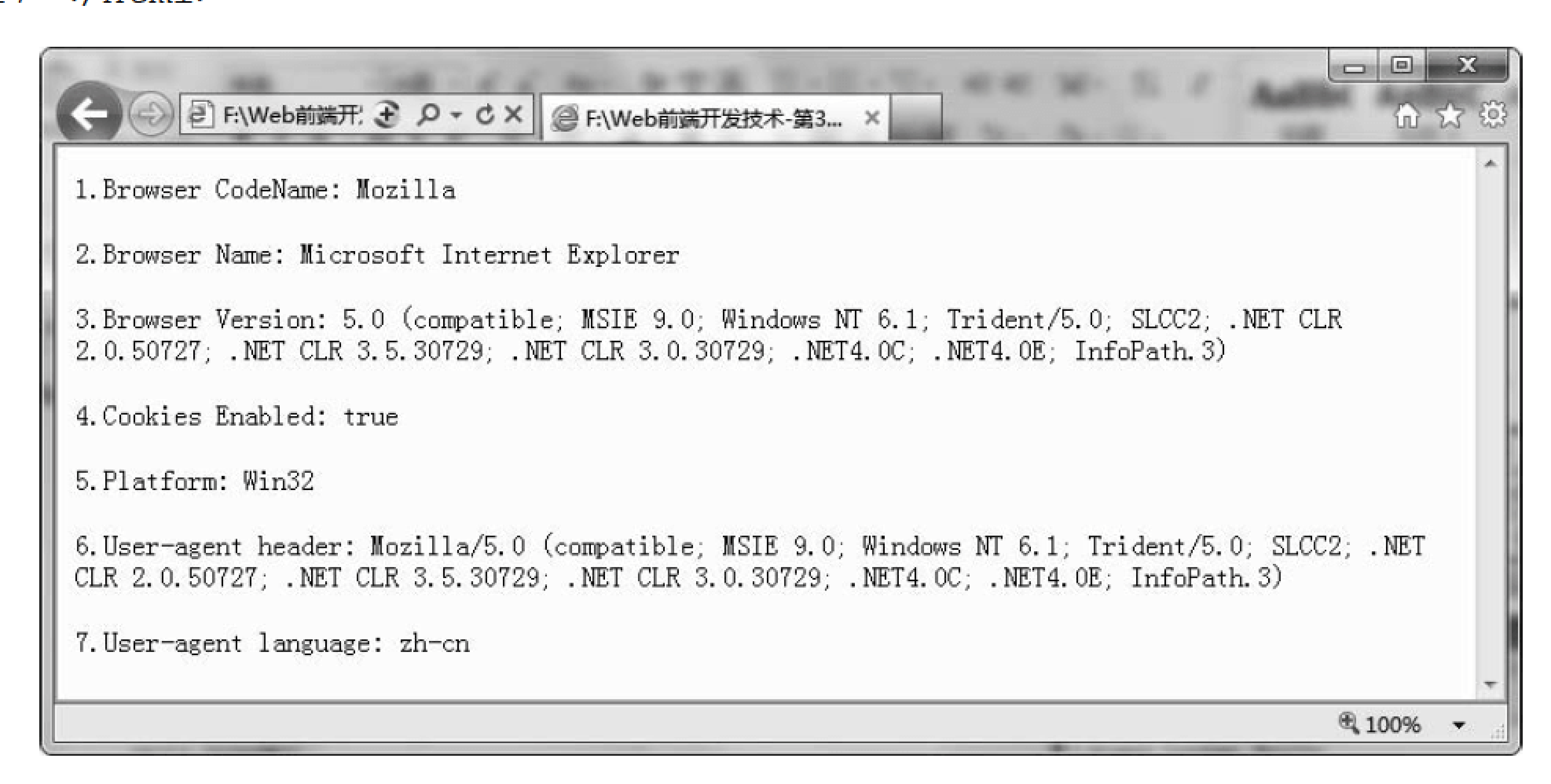



图 16-13　navigator 对象的应用

**代码解释**

代码中第 7～13 行获取浏览器对象的属性值给变量 txt 赋值，第 14 行通过 id 获取页面中的 div，将 txt 的值赋给 div 的 innerHTML 属性。

## 16.3.3　Screen 对象

screen 对象用于获取用户屏幕设置的相关信息，主要包括显示尺寸和可用颜色的数量信息。表 16-14 中给出 screen 对象常用的属性，这些属性得到了各种浏览器的普遍支持。

**表 16-14　screen 对象的属性及说明**

| 方　法　名 | 说　　明 | 方　法　名 | 说　　明 |
|---|---|---|---|
| availWidth | 返回可用的屏幕宽度 | height | 返回显示屏幕的高度 |
| availHeight | 返回可用的屏幕高度 | width | 返回显示屏幕的宽度 |

在浏览器窗口打开的时候，可以通过 screen 对象的属性来获取屏幕设置的相关信息。

【例 16-3-3】screen 对象的应用。代码如下所示，其页面效果如图 16-14 所示。

视频讲解

```
1 <!--  edu_16_3_3.html -->
2 <!doctype html>
3 <html lang="en">
4     <head>
5         <meta charset="UTF-8">
6         <title>screen对象的应用</title>
7         <script  type="text/javascript">
8         function getScreenInfo(){
9             document.write("<h3>screen对象的信息</h3><br>");
10            document.write("屏幕的总高度："+screen.height+"<br>");
11            document.write("屏幕的可用高度："+screen.availHeight+"<br>");
12            document.write("屏幕的总宽度："+screen.width+"<br>");
13            document.write("屏幕的可用宽度："+screen.availWidth+"<br>");
14        }
15        </script>
16    </head>
17    <body onload="getScreenInfo()">
18    </body>
19 </html>
```

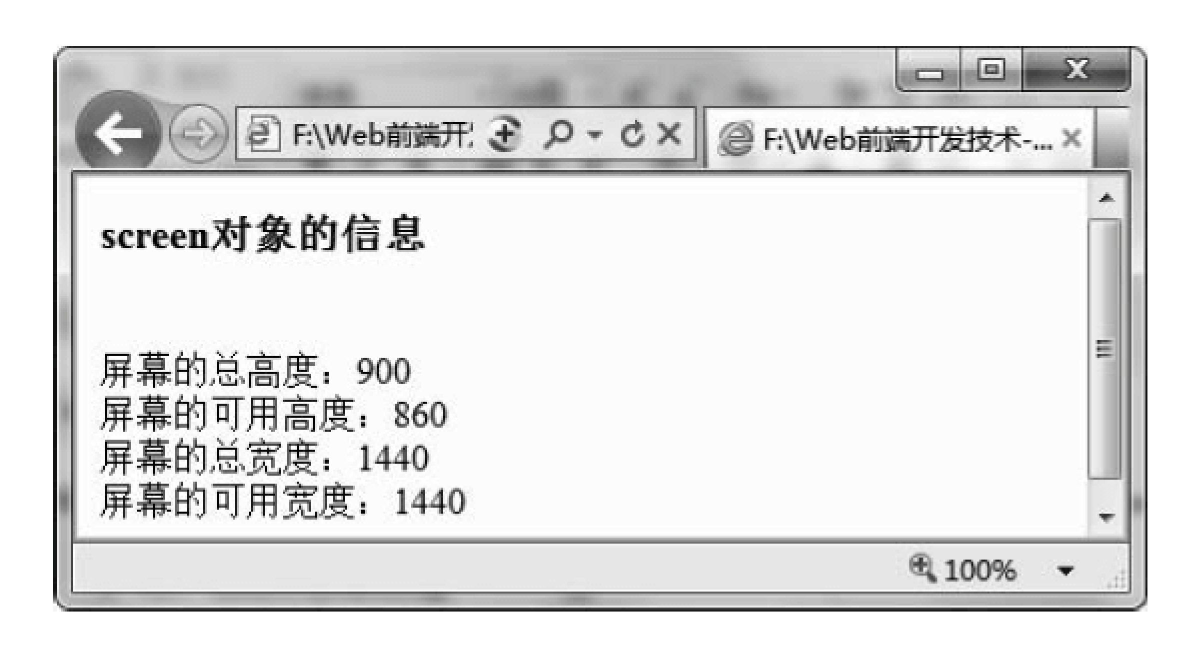



图 16-14　screen 对象的应用

**代码解释**

代码中第 8～14 行定义了一个 JavaScript 函数，名为 getScreenInfo()；第 17 行为 body 标记设置了 onLoad 事件句柄，当浏览器加载该页面时调用 getScreenInfo()，执行代码。

## 16.3.4　History 对象

history 对象表示窗口的浏览历史，并由 window 对象的 history 属性引用该窗口的 history 对象。history 对象是一个数组，其中的元素存储了浏览历史中的 URL，用来维护在 Web 浏览器的当前会话内所有曾经打开的历史文件列表。history 对象有三个常用的方法，如表 16-15 所示。

**表 16-15　history 对象的方法及说明**

| 方　法　名 | 说　　明 |
|---|---|
| forward() | 加载 history 列表中的下一个 URL |
| back() | 加载 history 列表中的前一个 URL |
| go(number\|URL) | 加载 history 列表中的某个具体页面。URL 参数指定要访问的 URL；number 参数指定要访问的 URL 在 history 的 URL 列表中的位置 |

history 对象的这三个方法与浏览器软件中的"后退"和"前进"按钮的功能一致。需要注意的是，如果没有使用过"后退"按钮或跳转菜单在历史记录中移动，而且 JavaScript 没有调用 history.back()或 history.go()方法，那么调用 history.forward()方法不会产生任何效果，因为浏览器已经处在 URL 列表的尾部，没有可以前进访问的 URL 了。在实际应用中的代码如下所示：

```
history.back()          //与单击浏览器后退按钮执行的操作一样
history.go(-2)          //与单击2次浏览器后退按钮执行的操作一样
history.forward()       //等价于单击浏览器前进按钮或调用history.go(1)
```

## 16.3.5 Location 对象

location 对象用来表示浏览器窗口中加载的当前文档的 URL，该对象的属性说明了 URL 中的各个部分，如图 16-15 所示。

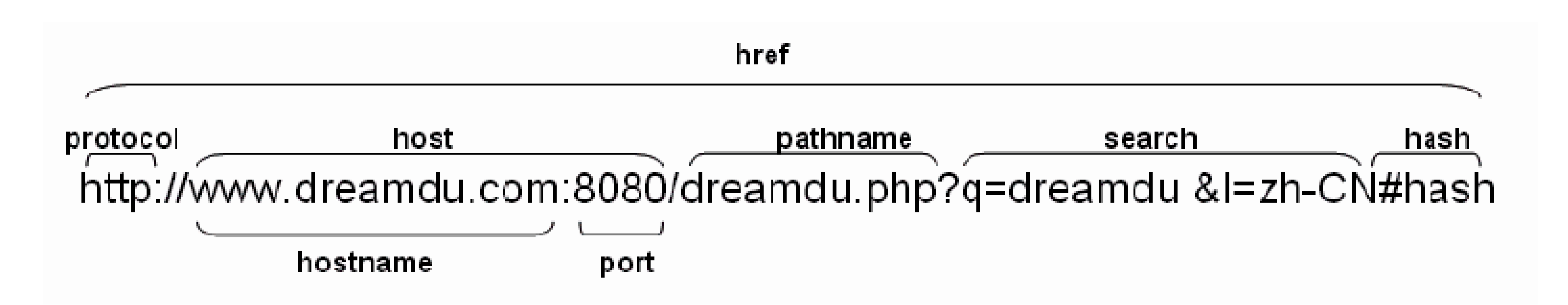



图 16-15 location 对象属性示意图

location 对象的常用属性如表 16-16 所示。

表 16-16 location 对象的属性及说明

| 属性名 | 说 明 | 属性名 | 说 明 |
|---|---|---|---|
| hash | 设置或返回从井号(#)开始的 URL(锚) | port | 设置或返回当前 URL 的端口号 |
| href | 设置或返回完整的 URL | pathname | 设置或返回当前 URL 的路径部分 |
| hostname | 设置或返回 URL 中的主机名 | host | 设置或返回 URL 中的主机名和端口号的组合 |
| protocol | 设置或返回当前 URL 的协议 | search | 设置或返回从问号(?)开始的 URL(查询部分) |

通过设置 location 对象的属性，可以修改对应的 URL 部分，而且一旦 location 对象的属性发生变化，就相当于生成了一个新的 URL，浏览器便会尝试打开新的 URL。虽然可以通过改变 location 对象的任何属性加载新的页面，但是一般不建议这么做，正确的方法是修改 location 对象的 herf 属性，将其设置为一个完整的 URL 地址，从而实现加载新页面的功能。

location 对象和 document 对象的 location 属性是不同的，document 对象的 location 属性是一个只读字符串，不具备 location 对象的任何特性，所以也不能通过修改 document 对象的 location 属性实现重新加载页面的功能。

location 对象除了上面所述的属性以外，还具有三个常用的方法，用于实现对浏览器位置的控制。location 对象的方法如表 16-17 所示。

表 16-17 location 对象的方法及说明

| 方 法 名 | 说 明 | 方 法 名 | 说 明 |
|---|---|---|---|
| reload() | 重新加载当前文档 | replace() | 用新的文档替换当前文档 |
| assign() | 加载新的文档 | | |

在实际应用中的代码如下所示：

```
location.assign("obj.html");//转到指定的URL资源
location.reload("obj.html");//加载指定的URL资源
location.replace("obj.html");//新的URL资源会替换当前的资源
```

【例 16-3-4】location 对象的应用。代码如下所示，页面效果如图 16-16 所示。

视频讲解

```
1 <!-- edu_16_3_4.html -->
2 <!doctype html>
3 <html lang="en">
4     <head>
5         <meta charset="UTF-8">
6         <script type="text/javascript">
7             function currLocation(){alert(window.location)}
8             function newLocation(){location.href="http://www. baidu. com"}
9         </script>
10    </head>
11    <body>
12        <input type="button" onclick="currLocation()" value="显示当前的
          URL">
13        <input type="button" onclick="newLocation()" value="改变 URL-百
          度">
14    </body>
15 </html>
```




图 16-16　location 对象的应用

上述代码中第 7 行、第 8 行定义了两个函数名分别为 currLocation()、newLocation()；第 12 行、第 13 行在 body 标记插入两个普通按钮，分别是“显示当前的 URL”和“改变 URL-百度”。当选择“显示当前的 URL”按钮时，通过告警消息框输出；当选择“改变 URL-百度”按钮时，在本窗口打开百度页面。

## 16.4　综 合 实 例

以“Web 前端开发技术”网络课程网站开发为例，设计一个含有二级水平导航菜单、图像切换、下拉列表导航等功能的网站，如图 16-17 所示。




图 16-17 Web 前端开发技术网络课程网站首页

**1. 页面布局设计**

根据如图 16-17 所示的页面效果设计页面布局，如图 16-18 所示。

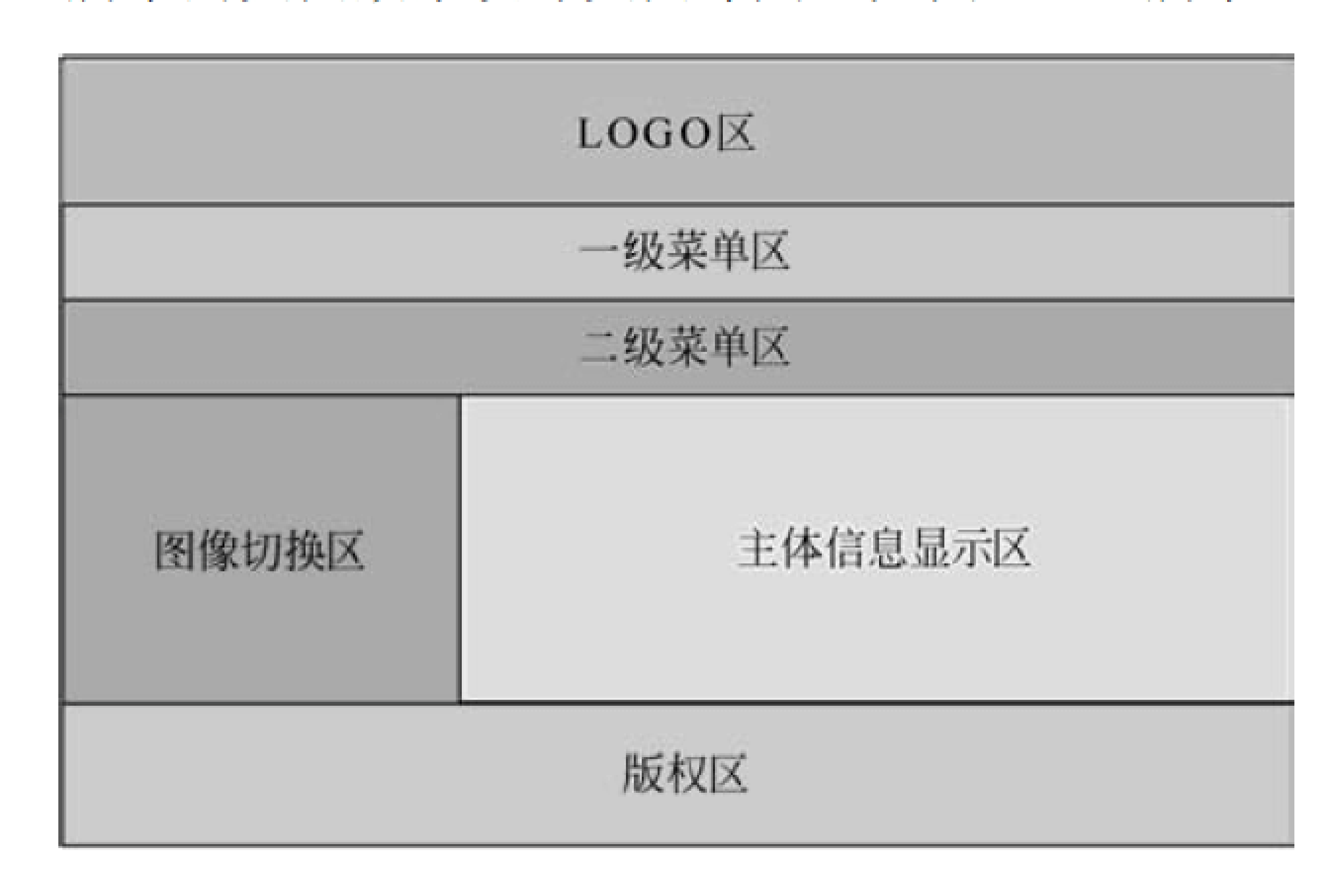



图 16-18 网站首页布局设计

**2. 网站中实现的主要技术**

- DIV+CSS+JavaScript 实现的二级导航菜单。
- JavaScript 实现图像自动定时切换。
- Window 对象 open 方法和 select 对象的 options、selecctedIndex 属性实现下拉列表导航功能。

1）JS 二级导航菜单

二级水平导航菜单实现技术分析：一级菜单、二级菜单在不同区域中单独显示；一级导航菜单采用无序列表实现，一个 li 标记表示一个主导航栏目，两个导航栏目之间插入一个分隔线（空 li 标记,设置背景图像）；在一级导航菜单上设置 onmouseover 事件句柄属性，绑定事件处理函数 qiehuan(num)；二级导航菜单显示规则：默认显示第一个一级导航栏目

对应的二级导航菜单，其余二级菜单默认是不显示，只有当鼠标悬停(盘旋)在相应的一级导航菜单上时才能调用 qiehuan(num)函数，将 id 为“qh_con”+num 的 div 的 display 属性值改为 block，显示其对应的二级导航菜单；所有的二级导航菜单分别定义在不同的 div 中。

一级导航菜单结构如下：

```
<ul id="nav">
   <li><a class="nav_on" id="mynav0" onmouseover="javascript:qiehuan(0)" href="web_first.html" target="framebody"><span>首 页</span></a></li>
   <li class="menu_line"></li>
   <li><a href="#" onmouseover="javascript:qiehuan(1)" id="mynav1" class="nav_off"><span>课程简介</span></a></li>
   <li class="menu_line"></li>
   <li><a href="#" onmouseover="javascript:qiehuan(2)" id="mynav2" class="nav_off"><span>主讲教师</span></a></li>
   …
</ul>
```

二级导航菜单统一放在一个 id 为“menu_con”的父 div 中，每一个二级子菜单单独放在一个独立的子 div 中。CSS 样式定义参见 style2menu.css，二级导航菜单结构如下：

```
<div id="menu_con">
   <div id="qh_con0" style="display: block">  <!-- 第一个子菜单 -->
      <ul>
         <li><a href="#"><span>课程发展</span></a></li>
         <li class="menu_line2"></li>          <!-- 子菜单分隔线 -->
         <li><a href="#"><span>课程特色</span></a></li>
         <li class="menu_line2"></li>          <!-- 子菜单分隔线 -->
         <li><a href="#"><span>教学成果</span></a></li>
      </ul>
   </div>
   <div id="qh_con1" style="display: block"> <!-- 第二个子菜单 -->
      …
   </div>
    ...          <!-- 第n个子菜单 -->
</div>
```

2）图像自动定时切换

图像自动定时切换实现技术分析：在一个 div 中插入一个图像的超链接，定义图像 img 标记的 id，通过 JavaScript 获取 img 标记对象，动态修改 img 的 src 属性，实现图像切换。使用 window 对象的 setInterval(code,interval)、clearInterval(intervalID)两个方法来实现时间执行代码和取消执行代码。在 switchpic.js 中定义初始化 init()、切换 switchPic()、重新鼠标移出 reStart()、鼠标悬停 pause()共四个 JavaScript 函数。

3）下拉列表框导航

下拉列表框导航实现技术分析：设置 select 标记的 onchange 事件句柄属性，并绑定事件代码，直接使用 window 对象的 open(url,name,features,replace)实现在单击下拉列表框中任一选项时，能够打开相关的超链接。代码如下：

```
<select size="1" name="d1"
 onchange="window.open(this.options[this.selectedindex].value)">
    <option>网络课程资源链接</option>
    <option value="http://www.icourses.cn/home/">中国mooc大学</option>
</select>
```

第 2 行设置 onchange=window.open(this.options[this.selectedindex].value)。第 4 行设置 option 标记的 value 属性指定目标 URL。其中列表框对象有 options(i)（返回列表框某一列表项）、selectedIndex（返回选中项的序号）属性。

### 3. 主要的实现代码

1）页面 HTML 代码 edu_16_4_1.html

```
<!doctype html>
<html lang="en">
    <head>
        <title>web前端开发技术课程网站</title>
        <meta charset="UTF-8">
        <meta name="keywords" content="html,css,javascript,web前端开发"/>
        <meta name="description" content="web前端开发技术，html、css、
        javascipt技术，开发综合实例" />
        <link href="style2menu.css" rel="stylesheet" type="text/css" />
        <script type="text/javascript" src="qiehuan.js"></script>
        <script type="text/javascript" src="switchpic.js"></script>
    </head>
<body onload="init();">
    <div id="container" class="">
        <div id="header" class="">
            <img src="images/web_logo.jpg" alt="">
        </div>
        <div id="menu_out">
            <div id="menu_in">
                <div id="menu">
                    <ul id="nav">
                    <li><a class="nav_on" id="mynav0" onmouseover=
                    "javascript:qiehuan(0)" href="web_first.html" target=
                     "framebody"><span>首 页</span></a></li>
                    <li class="menu_line"></li>
                    <li><a href="#" onmouseover="javascript:qiehuan(1)"
                    id="mynav1" class="nav_off"><span>课程简介</span></a>
                    </li>
                    <li class="menu_line"></li>
                    <li><a href="#" onmouseover="javascript:qiehuan(2)"
                    id="mynav2" class="nav_off"><span>主讲教师</span>
                    </a></li>
                    <li class="menu_line"></li>
                    <li><a href="#" onmouseover="javascript:qiehuan(3)"
                    id="mynav3" class="nav_off"><span>教学团队</span></a>
                    </li>
                    <li class="menu_line"></li>
                    <li><a href="#" onmouseover="javascript:qiehuan(4)"
                    id="mynav4" class="nav_off"><span>教学课件</span></a>
                    </li>
                    <li class="menu_line"></li>
                    <li><a href="#" onmouseover="javascript:qiehuan(5)"
                    id="mynav5" class="nav_off"><span>实验项目</span></a>
                    </li>
                    <li class="menu_line"></li>
                    <li><a href="#" onmouseover="javascript:qiehuan(6)"
                    id="mynav6" class="nav_off"><span>课程设计</span></a>
                    </li>
                    <li class="menu_line"></li>
                    <li><a href="#" onmouseover="javascript:qiehuan(7)"
```

```
            id="mynav7" class="nav_off"><span>在线测验</span></a>
            </li>
            <li class="menu_line"></li>
            <li><a class="nav_off" id="mynav8" onmouseover=
           "javascript:qiehuan(8)" href="#"><span>网络资源</span>
            </a></li>
        </ul>
        <div id="menu_con">
            <div id="qh_con0" style="display: block">
                <ul>
                    <li><a href="#"><span>课程发展</span></a></li>
                    <li class="menu_line2"></li>
                    <li><a href="#"><span>课程特色</span></a></li>
                    <li class="menu_line2"></li>
                    <li><a href="#"><span>教学成果</span></a></li>
                </ul>
            </div>
            <div id="qh_con1" style="display: none">
                <ul>
                    <li><a href="#"><span>教学大纲</span></a></li>
                    <li class="menu_line2"></li>
                    <li><a href="#"><span>教学计划</span></a></li>
                    <li class="menu_line2"></li>
                    <li><a href="web_practice.html" target=
                    "framebody">
                <span>实验计划</span></a></li>
                </ul>
            </div>
            <div id="qh_con2" style="display: none">
                <ul>
                    <li><a href="#"><span>教学工作</span></a></li>
                    <li class="menu_line2"></li>
                    <li><a href="#"><span>教学改革</span></a></li>
                    <li class="menu_line2"></li>
                    <li><a href="#"><span>科研成果</span></a></li>
                </ul>
            </div>
            <div id="qh_con3" style="display: none">
                <ul>
                    <li><a href="#"><span>储久良</span></a></li>
                    <li class="menu_line2"></li>
                    <li><a href="#"><span>姜  枫</span>
                    </a></li>
                    <li class="menu_line2"></li>
                    <li><a href="#"><span>王  巍</span>
                    </a></li>
                </ul>
            </div>
            <div id="qh_con4" style="display: none">
                <ul>
                    <li><a href="#"><span>HTML-PPT</span></a></li>
                    <li class="menu_line2"></li>
                    <li><a href="#"><span>CSS-PPT</span></a></li>
                    <li class="menu_line2"></li>
                    <li><a href="#"><span>JavaScript-PPT</span>
                    </a></li>
                </ul>
            </div>
```

```
                <div id="qh_con5" style="display: none">
                    <ul>
                        <li><a href="#"><span>HTML部分实验</span></a>
                        </li>
                        <li class="menu_line2"></li>
                        <li><a href="#"><span>CSS部分实验</span></a>
                        </li>
                        <li class="menu_line2"></li>
                        <li><a href="#"><span>JavaScript部分实验</span>
                        </a></li>
                    </ul>
                </div>
                <div id="qh_con6" style="display: none">
                    <ul>
                        <li><a href="#"><span>设计案例1</span></a></li>
                        <li class="menu_line2"></li>
                        <li><a href="#"><span>设计案例2</span></a></li>
                        <li class="menu_line2"></li>
                        <li><a href="#"><span>设计案例3</span></a>
                        </li>
                    </ul>
                </div>
                <div id="qh_con7" style="display: none">
                    <ul>
                        <li><a href="web_exam.html" target=
                        "framebody"><span>综合练习1</span></a></li>
                        <li class="menu_line2"></li>
                        <li><a href="web_exam.html" target=
                        "framebody"><span>综合练习2</span></a></li>
                        <li class="menu_line2"></li>
                        <li><a href="web_exam.html" target=
                        "framebody"><span>综合练习3</span></a></li>
                    </ul>
                </div>
                <div id="qh_con8" style="display: none">
                    <ul>
                        <li><a href="http://www.w3school.com.cn/
                        html/index.asp" target="framebody"><span>HTML
                        教程</span></a></li>
                        <li class="menu_line2"></li>
                        <li><a href="http://www.w3.org/TR/2014/REC-
                        html5-20141028/" target="framebody"><span>
                        HTML5规范</span></a></li>
                        <li class="menu_line2"></li>
                        <li><a href="http://www.php100.com/manual/
                        jquery/" target="framebody"><span> jQuery在线
                        手册</span></a></li>
                    </ul>
                </div>
                </div>
            </div>
            </div>
        </div>
        <div id="main"  >
            <div id="leftbar" class="">
                <a href="#"><img id="pic" src="images/example1.png"
                border="0" alt="" onmouseover="pause();"onmouseout=
                "reStart();"></a>
```

```
                </div>
            <div id="right" class="">
                <iframe name="framebody" src="web_first.html" ></iframe>
            </div>
        </div>
        <!-- 页面底部设计 -->
        <div class="bottom">
            <ul>
                <li><strong>友情链接：</strong>
                    <select size="1" name="d1" onchange="window.open
                    (this.options[this.selectedindex].value)">
                        <option>中国名牌大学</option>
                    <option value="http://www.tsinghua.edu.cn/">清华大学
                    </option>
                     <option value="http://www.pku.edu.cn/">北京大学
                     </option>
                     <option value="http://www.fudan.edu.cn/">复旦大学
                     </option>
                     <option value="http://www.sjtu.edu.cn/">上海交通大学
                     </option>
                     <option value="http://www.xjtu.edu.cn/">西安交通大学
                     </option>
                    </select>
                <select size="1" name="d1" onchange="window.open(this.
                options[this.selectedindex].value)">
                    <option>网络课程资源链接</option>
              <option value="http://www.icourses.cn/home/">中国mooc大学
              </option>
               <option value="http://www.jingpinke.com/">国家精品课程共享
               服务信息平台</option>
                <option value="http://jpkc.fudan.edu.cn/">复旦大学精品课程
                </option>
                    <option value="http://jpkc.nwu.edu.cn/course.
                    php#xb_jpkc">西北大学精品课程建设网</option>
                    <option value="http://www.cncourse.com/portal/
                    indexdefault">中国高校课程网</option>
                    <option value="http://www.intel.com/cn/index.htm">中
                    国教育和科研计算机网</option>
                </select>
                </li>
            <li>
            web前端开发技术课程建设小组 2015-2020&copy;保留所有权利,未经允许不
            得复制、镜像</li>
            </ul>
        </div>
        </div>
    </body>
</html>
```

2）二级导航菜单切换显示 qiehuan.js 代码

```
/* 切换菜单显示 qiehuan.js */
function qiehuan(num){
    for(var id = 0;id<=8;id++){
        if(id==num) {
            document.getElementById("qh_con"+id).style.display="block";
            document.getElementById("mynav"+id).className="nav_on";
```

```
        } else {
            document.getElementById("qh_con"+id).style.display="none";
            document.getElementById("mynav"+id).className="";  }
    }
}
```

3）图像切换 switchpic.js 代码

```
/* switchpic.js */
//定义全局变量
var CurScreen=1;          //当前显示的图像
var MaxScreen=5;          //最多可切换图像数
var timer=null;           //定时器变量
function $(id){return document.getElementById(id);}
function switchPic(){     //切换图像函数，定时触发
    if (CurScreen==MaxScreen){CurScreen=1;    }else{CurScreen++;}
    //切换图像到最大值时返回1
    $("pic").src="images/example"+CurScreen+".png";//更换图像的文件名
}
function reStart(){       //重新开始，鼠标移出时触发
    switchPic();          //切换下一张图
    init();               //开始定时器
}
function pause(){         //暂停切换，鼠标悬停时触发
    clearInterval(timer); //清除定时器
}
function init(){          //初始化函数，在body加载时触发
    timer=setInterval('switchPic();',1000);
}
```

4）页面 CSS 样式文件 style2menu.css 代码

```
/* JS+DIV+CSS 二级导航菜单 style2menu.css */
@charset "utf-8";
/*全局样式*/
*{font-size:12px;color:#666666;
    font-family: "宋体",Arial, Helvetica, sans-serif;    }
body{margin:0px auto;padding:0px;text-align:center;}
#container{width:960px;padding:0 auto;margin:0 auto;}
img{width:960px;height:160px;}
/*主导航菜单*/
#menu ul{
    padding:0;    border:0;
    list-style:none;    line-height:150%;
    margin-top: 0;    margin-right: 0;
    margin-bottom: 0;    margin-left: 40px;
}
#menu_out{
    width:960px;    padding-left:4px;
    margin-left:auto;    margin-right:auto;
    background:url("images/menu_left.gif") no-repeat left top;
    overflow:hidden; /* 溢出部分隐藏 */
}
#menu_in{
    background:url("images/menu_right.gif") no-repeat right top;
```

```
    padding-right:4px;
}
#menu{
    background:url("images/menu_bg.gif") repeat-x;
    height:73px;width:960px;
}
.menu_line{
    background:url("images/menu_line.gif") no-repeat center top;
    width:8px;
}
.menu_line2{
    background:url("images/menu_line2.gif") no-repeat center top;
    width:15px;
}
#nav{ padding-left:20px;width:960px;}
#nav li{float:left;   height:35px;}
#nav li a{
    float:left;   display:block;   padding-left:6px;
    height:35px;cursor:pointer;   text-decoration:none;
    background:url("images/menu_on_left.gif") no-repeat left top;
}
#nav li a span{
    float:left;   padding:11px 14px 10px 10px;
    line-height:14px;text-decoration:none;
    background:url("images/menu_on_right.gif") no-repeat right top;
    font-size:14px;   font-weight:bold;   color:#FFFFFF;
}
#nav li .nav_on{  /*鼠标经过时变换背景，方便JS获取样式*/
    background-position:left 100%;
}
#nav li .nav_on span{  /*鼠标经过时变换背景，方便JS获取样式*/
    background-position:right 100%;
    color:#333333;text-decoration:none;
    padding:14px 14px 7px 10px;
}
/*子栏目*/
#menu_con{
    text-align:left;   padding-left:20px;   clear:both;
}
#menu_con li{
    float:left;   height:22px;margin-top:8px;
}
#menu_con li a{
    display:block;   float:left;
    background:url("images/menu_on_left2.gif") no-repeat left top;
    cursor:pointer;   padding-left:3px;
}
#menu_con li a span{
    float:left;   padding:6px 10px 4px 10px;line-height:12px;
    background:url("images/menu_on_right2.gif") no-repeat right top;
}
#menu_con li a:hover{
    text-decoration:none;
    background:url("images/menu_on_left2.gif") no-repeat left bottom;
}
#menu_con li a:hover span{
    background:url("images/menu_on_right2.gif") no-repeat right bottom;
```

```
81 }
82 #main{width:960px;height:300px;}
83 #leftbar{width:298px;height:298px;float:left;
84     border:1px solid #F1F1F1;}
85 #leftbar img{width:298px;height:298px;}
86 #right{width:660px;height:300px;float:left;}
87 #right iframe{width:660px;height:298px;border:0px;padding:0px;margin:
   0px;}
88 .bottom{clear:both;height:80px;
89     background:#FF9820;text-align:center;padding-top:20px;
90     color:white;font-size:18px;width:960px;
91 }
92 .bottom ul{list-style:none;color:#FFEEDD;}
```

上述代码中第 6 行设置 padding:0px 和 margin:0px auto 可以保证在不同的浏览器中显示效果相同，因为有些浏览器默认的顶部或左右空白。第 7 行设置容器的样式，可以保证在不同浏览器和不同分辨率的设备显示效果相同。第 9～81 行设置主导航和二级导航区的样式，其中第 20 行设置 overflow 属性，主要解决 IE 浏览器溢出部分的显示问题。第 82～87 行设置主体区的样式。第 88～92 行设置版权区的样式。

最后利用 MultiBrowser 软件检测网页在不同浏览器中的兼容性。通过 View 打开所设计的 Web 前端开发技术网络课程网站，可以看到在 Chrome18、Firefox3.6、Firefox11、IE 7、IE8 等浏览器中网站的显示效果相同，如图 16-19 所示。




图 16-19　网站在多浏览器兼容性测试软件中页面效果

# 本 章 小 结

本章介绍了 JavaScript 对象的概念及 Array、Date、Math、Number、String、Boolean 等常用的核心对象。通过大量的示例讲解了在实际开发中如何运用这些对象的方法和属性。

HTML 文档中的每个标记都是一个节点，这些标记之间存在着一定的关系，这种描述页面标记关系的树形结构称为 DOM 节点树。对于 DOM 节点的访问除了通过 form 对象的 elements 属性或该节点的 name 属性来访问外，还可以通过 document 对象的 getElementById()、getElementsByName()、getElementsByTagName()等方法来访问；document 对象应用非常广泛，除了访问节点外，还可以调用该对象的方法和属性来动态地创建和修改节点、设置节点的属性。

BOM 定义了浏览器对象（window、history、document、location、screen、navigator、frame 等对象）的组成和相互关系，描述了浏览器对象的层次结构。在 BOM 中，每个对象都含有若干属性和方法，使用这些属性和方法可以操作 Web 浏览器窗口中的不同对象，控制和访问 HTML 页面中的不同内容。

# 练习与实验

## 练习 16

**1．选择题**

（1）定义 JavaScript 数组方法正确的是（　　）。

A．var arrayList ={"cat" , "dog" , "monkey"}

B．var arrayList =new Array{"cat" , "dog" , "monkey"}

C．var arrayList =new Array("cat" , "dog" , "monkey")

D．var arrayList =new Array["cat" , "dog" , "monkey"]

（2）利用下标来访问数组时，最小下标是从（　　）开始的。

A．0　　B．1　　C．–1　　D．2

（3）求 3 和 5 中的最小数正确的函数是（　　）。

A．Math.min(3,5)　　B．Math.ceil(3,5)

C．Math.max(3,5)　　D．min(3,5)

（4）以下选项中，可以获得值为 false 的 Boolean 对象的是（　　）。

A．var a = new Boolean(1)　　B．var a = new Boolean("abc")

C．var a = new Boolean(true)　　D．var a = new Boolean( )

（5）下列不属于访问指定节点的方法的是（　　）。

A．obj.value　　B．getElementsByTagName()

C．getElementsByName()　　D．getElementById()

（6）能够创建元素节点的方法的是（　　）。

A．createElement()　　B．getElementById()

C．getElementByName()　　D．forms.length

（7）下列代码分析正确的是（　　）。

```
1 function createNode(){
2     var p1=document.createElement("p");
3     var txt=document.createTextNode("Hello!");
4     p1.appendChild(txt);
5     document.appendChild(p);
6 }
```

A．代码第 2 行是创建一个<p>元素标记

B．代码第 4 行是为文档添加文本节点

C．<p>是文本节点的子节点

D．函数的功能是创建新的文本节点

（8）在告警消息框中输出“hello world!”信息正确的是（　　）。

A．alertBox("hello world!")　　B．msgBox("hello world!")

C．alert("hello world!")　　D．alertMsg("hello world!")

（9）下面这两行代码的功能是（　　）。

```
1 <a herf="javascript:history.back()"></a>
2 <a herf="javascript:history.forward()"></a>
```

A．代码第 1 行的功能相当于后退按钮

B．代码第 2 行的功能相当于后退按钮

C．代码第 1 行的功能相当于前进按钮

D．以上表述都不正确

（10）对 location 对象的 href 属性的叙述错误的是（　　）。

A．可以获取当前路径　　B．可以改变当前路径

C．可以用来刷新页面　　D．是只读属性

（11）使用 location 对象的（　　）方法可以实现用新 URL 取代当前窗口的 URL。

A．load　　B．onload　　C．replace　　D．open

**2．填空题**

（1）可以通过 Array 对象的______属性来获得数组的长度。

（2）使用 Math 对象的________方法可以获得 0～1 之间的随机数，使用 Math 对象的______属性可以获得圆周率。

（3）在 JavaScript 中，Boolean 对象只有两种状态，分别是______和______。

（4）DOM 是__________________________的英文缩写，一个最基本的 DOM 树通常由三种类型的节点组成，分别是____________、_____________和______________。

（5）document 对象中包含了三个访问文档节点的方法，这三个方法分别是______、______和______。

（6）document 对象包含一些创建和修改节点的方法，如可以通过调用 document 对象的__________________________方法来创建一个元素节点，通过调用 document 对象的______________________方法来删除一个子节点，通过调用 document 对象的____________________方法来添加一个子节点。

（7）使用 document 对象＿＿＿＿＿＿和＿＿＿＿＿＿属性可以获取节点的内容。

（8）浏览器对象模型（BOM）主要包含＿＿＿＿＿、＿＿＿＿＿、＿＿＿＿＿、＿＿＿＿＿＿、＿＿＿＿＿、frame 和 document 共七个对象，＿＿＿＿＿对象是最顶层对象。

（9）在实际的开发中，使用 window 对象的＿＿＿＿＿方法可以产生确认框，使用 window 对象的＿＿＿＿方法可以产生告警框，使用 window 对象的＿＿＿＿方法可以产生提示框。

（10）＿＿＿＿＿＿＿对象用于获取用户浏览器的相关信息。

**3．简答题**

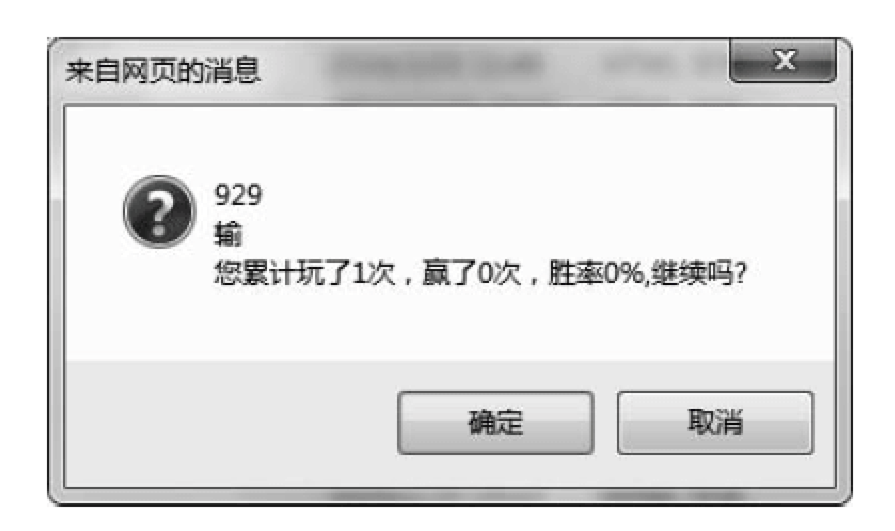
（1）什么是 document 对象？如何获取文档对象上的元素？

（2）什么是浏览器对象模型？它包含哪些对象？

（3）简述 window 对象有哪些常用的属性和方法。

## 实验 16

1．设计模拟幸运数字机游戏。设幸运数字为 8，每次由计算机随机生成三个 1～9 之间的随机数，当这三个随机数中有一个数字为 8 时，就算赢了一次，如图 16-20 所示。



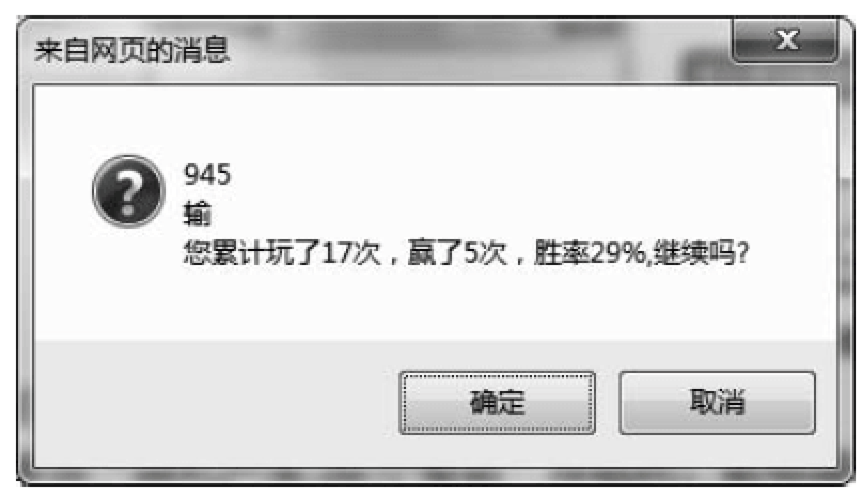




图 16-20　幸运数字机游戏页面

2．按如图 16-21 所示的布局，完成下列功能：

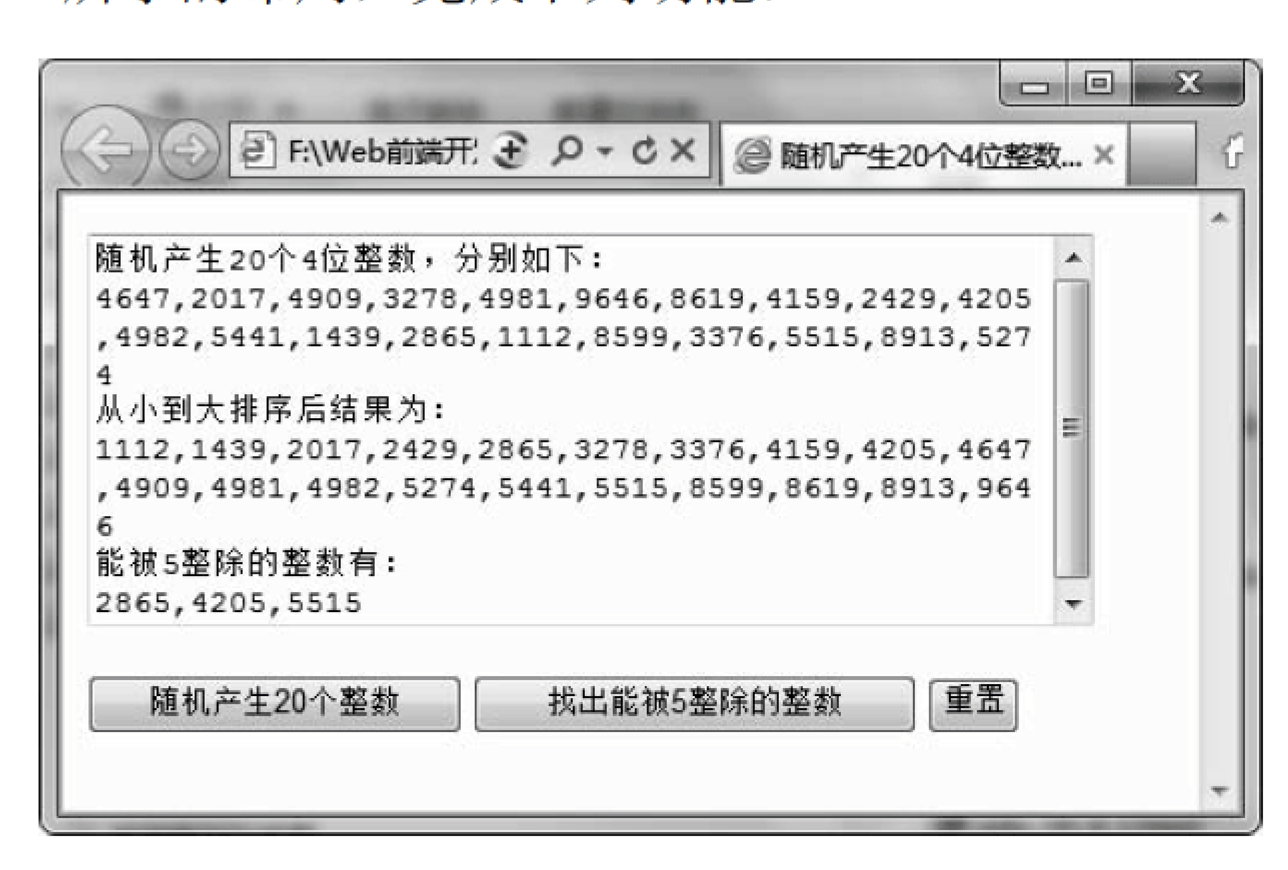



图 16-21　随机产生批量整数、排序、找特征数

（1）单击“随机产生 20 个整数”按钮时，能够随机产生 20 个 4 位整数（1000～9999），并将产生的 20 个整数写入到数组中，将其从小到大进行排序，输出在多行文本框中；

（2）单击“找出能被 5 整除的整数”按钮时，从产生的 20 个随机整数中找出能够被 5 整除的整数，并在多行文本框中输出；

（3）单击“重置”按钮时，将多行文本框中的所有内容清空。

# 第 17 章　HTML5 高级应用

本章学习目标

HTML5 高级应用主要涉及需要借助于 JavaScript 脚本才能实现的功能，例如 Web 存储、画布 Canvas、拖放、Web Worker 等。学会利用这些新特性解决实际工程中应用问题，提高 Web 页面的用户体验度，改善交互界面。

Web 前端开发工程师应掌握以下内容:

- 学会使用 Web 本地存储对象解决客户端数据存储问题。
- 掌握 Canvas 基本语法和学会绘制各种图形、文字及图像。
- 学会使用 Web 拖放技术解决简单的实际应用问题。
- 理解 Web Worker 多线程工作原理，学会使用多线程解决简单的实际应用问题。

## 17.1　HTML5 Web Storage

HTML5 提供了两种在客户端存储数据的新方法，分别是持久化的数据存储 localStorage、会话式的数据存储 sessionStorage。HTML5 之前客户端数据存储是由 cookie 完成的，由于 cookie 不能存储大量数据，需要通过服务器的请求来传递，往往造成 cookie 响应速度慢、效率低。在 HTML5 中，数据不需要由每个服务器进行请求传递，只需在有请求时使用数据，这样就不会影响网站的性能，而且能够存储大量数据。数据通常以键值对(key-value pair)形式存在，Web 网页的数据只允许该网页访问使用。

### 17.1.1　localStorage 对象

localStorage 对象存储的数据没有时间限制，所以称为持久化存储，数据存储长期可用。使用此类对象之前，最好先检查一下浏览器是否支持。检查代码如下：

```
if(typeof(Storage)!=="undefined") {
   //是的! 支持 localStorage sessionStorage对象! //一些代码...}
else { //抱歉! 不支持web存储。 }
```

localStorage 对象和 sessionStorage 对象具有同样的方法，仅仅是对象名称不同而已。下面列出 localStorage 对象的常用方法。

- localStorage.setItem(key,value)：保存数据。
- localStorage.getItem(key)：读取数据。
- localStorage.removeItem(key)：删除单个数据。
- localStorage.clear()：删除所有数据。

- localStorage.key(index)：得到某个索引的 key。

【例 17-1-1】localStorage 对象的应用。代码如下，页面效果如图 17-1 所示。

视频讲解

```
<!-- edu_17_1_1.html -->
<!DOCTYPE html>
<html>
    <head>
        <meta charset="UTF-8">
        <title>localStorage对象的应用</title>
        <style type="text/css">
            div{text-align:center;padding:20px;border:10px ridge
            #005A9C;width:350px;height:250px;margin:0 auto;}
        </style>
    </head>
    <body>
        <div>
            <h3>最可爱的人评选</h3>
            <img src="45567.jpg" width="80" height="80" border="0"
            alt=""><br>
            <p id="result"></p>
            <p>刷新页面票数会增长。</p>
            <p>请关闭浏览器后重试仍会增长</p>
        </div>
        <script type="text/javascript">
            var tickets=0;
            localStorage.setItem(tickets,0);
            if (localStorage.tickets){
                localStorage.tickets=parseInt(localStorage.tickets)
                +1;    }
            else{    localStorage.tickets=1;    }
            document.getElementById("result").innerHTML="已投: "+
            localStorage.tickets + "票";
        </script>
    </body>
</html>
```

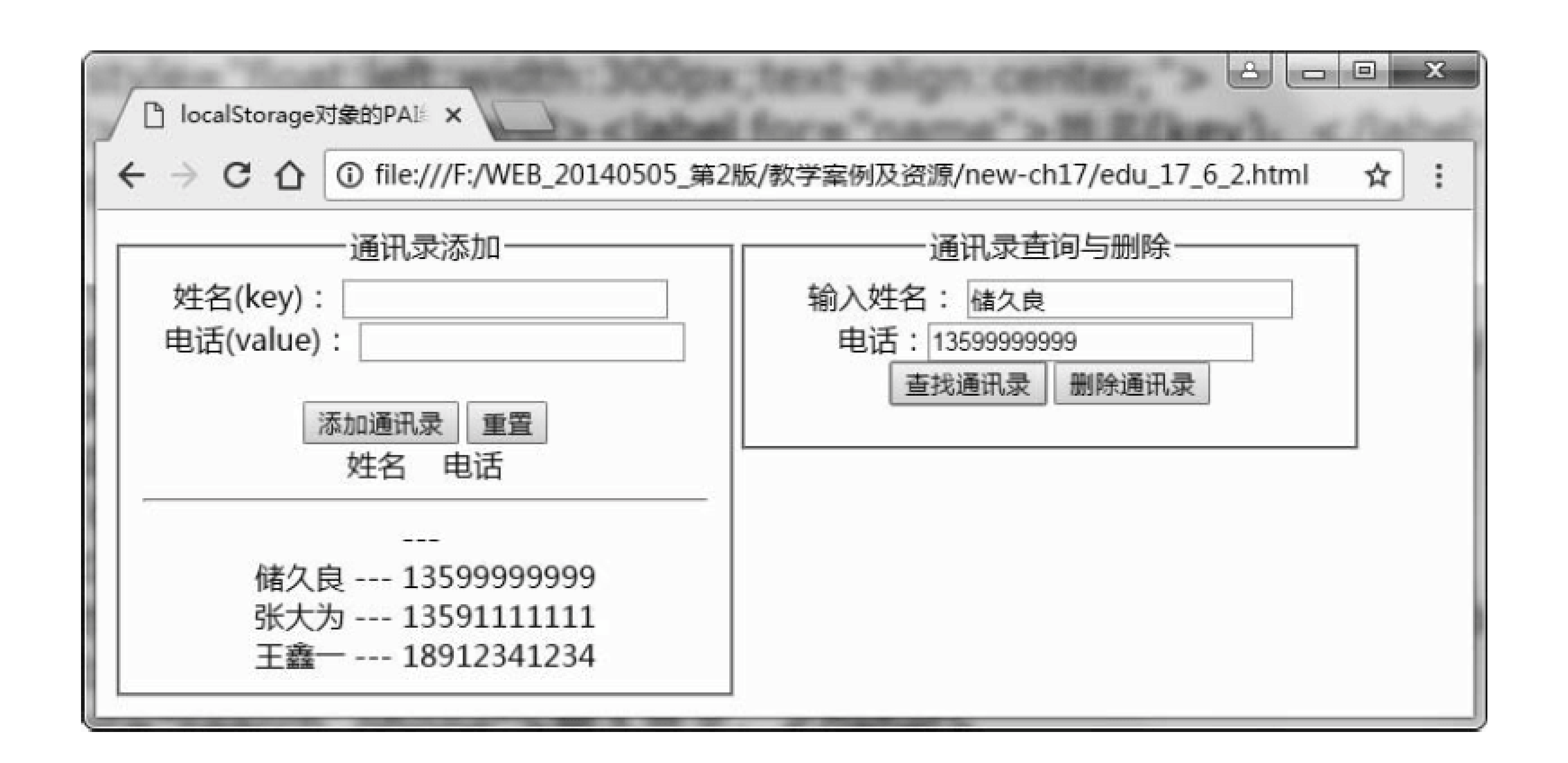



图 17-1　HTML5 localStorage 对象的应用

## 17.1.2　sessionStorage 对象

sessionStorage 对象针对一个 session 进行数据存储。数据存储周期短，当用户关闭浏览器窗口后，数据会被删除。该对象的方法与 localStorage 对象方法相同。

【例 17-1-2】localStorage 对象实现的简易通讯录。代码如下，页面效果如图 17-2 所示。




视频讲解

图 17-2　localStorage 对象实现的简易通讯录

```
<!-- edu_17_1_2.html -->
<!DOCTYPE html>
<html>
    <head>
    <meta charset="UTF-8">
    <title>localStorage对象的API综合应用</title>
    <script>
        //载入所有存储在localStorage的通讯录信息
        loadAllInfo();
        //保存一条通讯记录数据，同时显示在div中
        function $(id){return document.getElementById(id);}
        function saveInfo(){
            var name1=$("username").value;     //取姓名
            var phone1=$("userphone").value;   //取电话
            if (name1!="" && phone1!="")       //不为空处理
            {
                localStorage.setItem(name1,phone1);
                loadAllInfo();
                alert("添加成功");
            }else{                            //姓名或电话为空，告警并获得焦点
                alert("请输入姓名和电话！");
                $("username").focus();
            }
        }
        //以姓名查找通讯录信息
        function findForName(){
            var searchname = $("search_name").value;
            var searchphone = localStorage.getItem(searchname); //取电话
            $("userphone1").value=searchphone;                //填充电话
        }
        //从localStorage中取出所有通讯录中信息，并展现到界面上
        function loadAllInfo(){
            //localStorage.clear();
            var result = "";
            if(localStorage.length>0){
                result += "姓名    电话</br><hr>";
                for(var i=0;i<localStorage.length;i++){
                    var name= localStorage.key(i);
                    var phone = localStorage.getItem(name);
                    result += name+"  ---   "+phone+"</br>";
                }
```

```
                $("displayallinfo").innerHTML = result;
            }else{
                $("displayallinfo").innerHTML = "数据为空……";
            }
        }
        //删除某一条通讯信息
        function deleteName(){
            localStorage.removeItem($("search_name").value);   //取电话
            $("search_name").value="";                           //填充电话
            loadAllInfo();
        }
    </script>
    </head>
    <body>
        <fieldset style="float:left;width:300px;text-align:center;">
            <legend>通讯录添加</legend><label for="name">姓名(key):</label>
            <input type="text" id="username" name="username" required/><br/>
            <label for="telphone">电话(value): </label>
            <input type="text" id="userphone" name="userphone" required/><br/>
            <br><input type="button" onclick="saveInfo()" value="添加通讯录"/>
            <input type="reset">
            <div id="displayallinfo" name="displayallinfo"></div>
        </fieldset>
        <fieldset style="float:left;width:300px;height:100px;text-align:center;">
            <legend>通讯录查询与删除</legend>
            <label for="search_phone">输入姓名: </label>
      <input type="text" id="search_name" name="search_name" required/><br>
      <label>电话: </label><input type="text" name="" id="userphone1" readonly>
            <br><input type="button" onclick="findForName()" value="查找通讯录"/>
            <input type="button" onclick="deleteName()" value="删除通讯录"/>
        </fieldset>
    </body>
</html>
```

### 17.1.3 浏览器端数据库 IndexedDB

简单来说，IndexedDB 是一种轻量级 NoSQL（Not Only SQL，泛指非关系型）数据库，用来持久化大量（250MB）客户端数据。它可以让 Web 应用程序具有非常强大的查询能力，并且可以离线工作。IndexedDB 的数据操作直接使用 JS 脚本，不依赖 SQL 语句（最初的 Web SQL 数据库已被废弃），操作返回均采用异步。而 localStorage 和 sessionStorage 对象是采用同步技术实现少量（2.5～10MB）客户端数据（字符串）存储。一个网站可能有一个或多个 IndexedDB 数据库，每个数据库必须具有唯一的名称。WebStorage 可以用来存储键值对（key-value pair），然而它无法提供按顺序检索、高性能地按值查询或存储重复的键的功能。

使用 IndexedDB 的基本步骤如下所示：

- 打开数据库并且开始一个事务。
- 创建一个对象仓库(Object Store)。
- 构建一个请求来执行一些数据库操作，例如增加或提取数据等。
- 通过监听正确类型的 DOM 事件以等待操作完成。
- 在操作结果上进行一些操作（可以在 request 对象中找到）。

**1．浏览器支持 IndexedDB 数据库情况判断**

由于 IndexedDB 的规范尚未最终定稿，不同的浏览器厂商使用不同的浏览器前缀实现 IndexedDB API，例如基于 Gecko 内核的浏览器使用 moz 前缀，基于 WebKit 内核的浏览器使用 webkit 前缀。为了能够支持多种厂家的浏览器，建议采用 window.indexedDB 来判断浏览器是否支持 IndexedDB 数据库。代码如下所示：

```
var indexedDB=window.indexedDB || window.mozIndexedDB ||
  window.webkitIndexedDB;   //获得IndexedDB对象
  if(window.indexedDB){
      alert("您的浏览器支持IndexedDB数据库。");
  }else{
      alert("您的浏览器不支持IndexedDB数据库。");
  }
```

**2．数据库创建与打开**

使用 window.indexedDB.open(DBName, DBVersion)来打开数据库。语法如下：

```
var request=window.indexedDB.open(DBName, DBVersion);
//数据库存在打开它；否则创建
```

若 DBName 数据库创建之前并不存在，则会调用 onupgradeneeded 接口，在这个函数中可以进行数据库初始化和创建索引。

【例 17-1-3】创建名为 myBooks 的数据库，并创建名为 books（ObjectStore 相当于表）的数据仓库，为数据仓库添加两个对象（两本图书）数据。代码如下所示，数据库和对象仓库建立情况如图 17-3 所示。

视频讲解

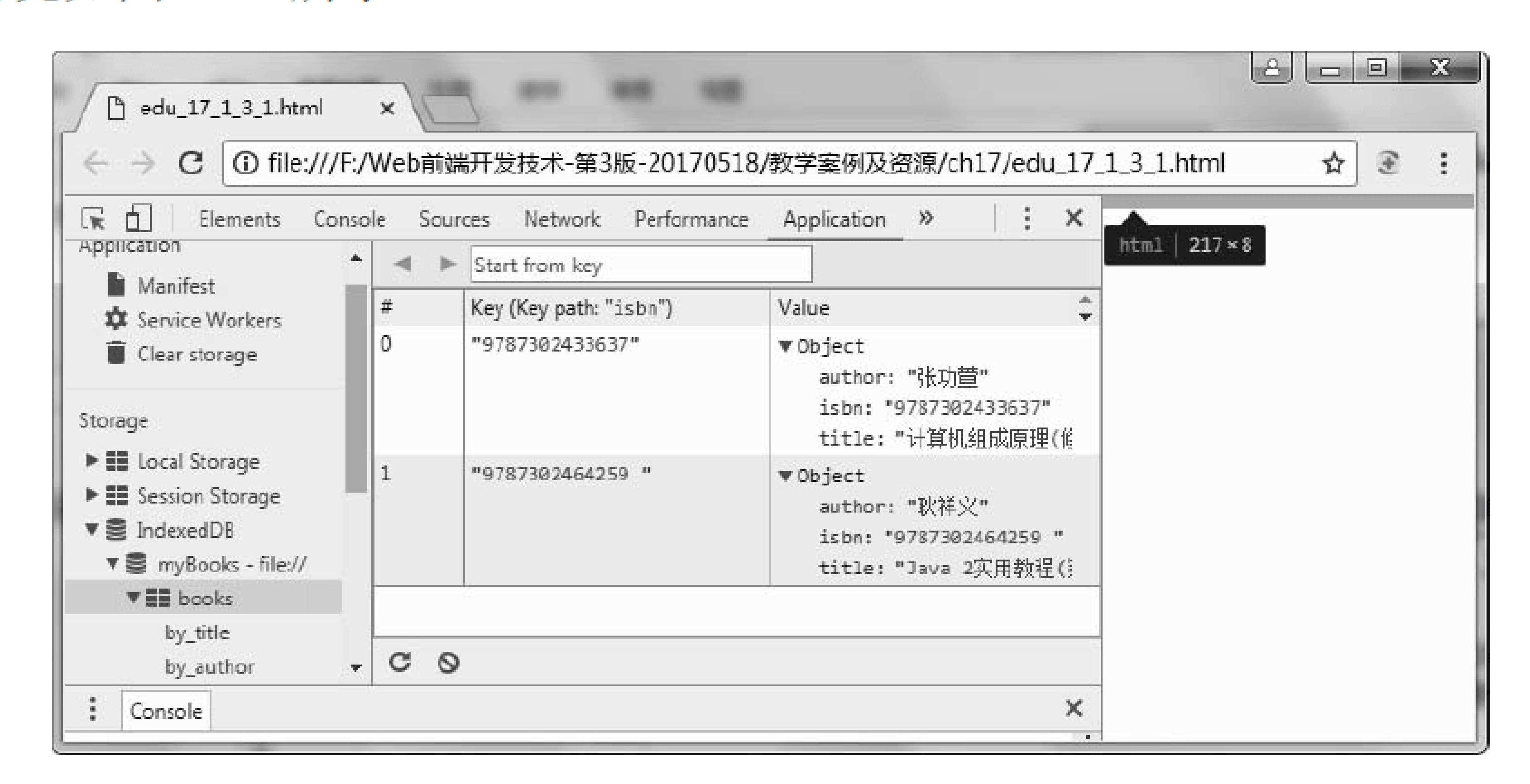



图 17-3　创建 myBooks 数据库和 books 对象仓库的结构图

```
var request=window.indexedDB.open("myBooks",1);
```

```
                                              //数据库存在，则打开它；否则创建
request.onerror = function(event) {      //捕获连接失败事件，并处理
   alert("数据库连接失败："+event.target.errorCode);        //提示错误信息
}
request.onupgradeneeded = function(event) {
  //当此数据库创建前不存在时，进行初始化
  var db = request.result;
  var store = db.createObjectStore("books", {keyPath: "isbn"});
  var titleIndex = store.createIndex("by_title", "title", {unique:
  true});                                  //标题索引
  var authorIndex = store.createIndex("by_author", "author");//作者索引
  //填入初始值，添加2本书信息
  store.put({title:"计算机组成原理(修订版)",author:"张功萱", isbn:
 "9787302433637"});
  store.put({title:"Java 2实用教程(第5版)", author: "耿祥义", isbn:
  "9787302464259 "});
}
request.onsuccess = function(event) {   //捕获连接成功事件，并处理
  db=event.target.result;    //连接成功时，获取数据库对象(也可用request.result)
  alert("数据库连接成功");
}
```

在连接到数据库后，request 会监听三种状态。

- success：打开或创建数据库成功后绑定指定函数。
- error：打开或创建数据库失败后绑定指定函数。
- upgradeneeded：更新数据库后绑定指定函数。

upgradeneeded 状态是在 indexedDB 创建新的数据库时和 indexedDB.open（DBName, DBVersion）DBVersion（数据库版本号）发生变化时才能监听到此状态。当版本号不发生变化时，不会触发此状态。数据库的 ObjectStore 的创建、删除等都是在这个监听事件下执行的。

需要注意的有两点：

（1）当数据库连接时，open()方法返回一个 IDBOpenDBRequest 对象，调用函数定义在这个对象上。

（2）在连接建立成功时，会触发 onsuccess 事件，调用函数接收一个事件对象 event 作为参数，其 target.result 属性指向打开的 IndexedDB 数据库。也可以使用监听器来捕获 3 个事件，分别为 success、error、upgradeneeded。可以通过下列方法为页面元素（对象）添加事件监听器。代码如下所示：

```
element.addEventListener(event, function, useCapture);
```

代码中 addEventListener()方法有三个参数。第 1 个参数为 event，为某元素指定监听事件名称，如 success、click 等，而不是 onsuccess、onclick 等事件句柄。第 2 个参数为 function，为某个事件绑定（指派）事件处理函数。第 3 个参数 useCapture，可选，布尔值，指定事件是否在捕获或冒泡阶段执行。其值为 true 表示事件句柄在捕获阶段执行；默认值为 false 表示事件句柄在冒泡阶段执行。

indexedDB 对象的 Open()方法需要监听的事件代码如下所示：

```
request.addEventListener('success', function(event){ //打开或创建数据库成功
}, false);                              //第3个参数为false:表示在冒泡阶段执行
request.addEventListener('error', function(event){ //打开或创建数据库失败
}, false);                              //第3个参数为false:表示在冒泡阶段执行
request.addEventListener('upgradeneeded', function(event){
                                        //更新数据库时执行
}, false);                              //第3个参数为false:表示在冒泡阶段执行
```

**3．创建与删除 ObjectStore**

ObjectStore（对象仓库，又称对象存储空间）是 IndexedDB 数据库的基础，在 IndexedDB 中并没有关系型数据库中的表，而是使用对象仓库（相当于关系型数据库的表）来存储数据。

1）createObjectStore()方法创建对象仓库

```
var store=db.createObjectStore(storeName,{keyPath: primaryKey,
autoIncrement: true|false});//keyPath称为键路径，作为ObjectStore的搜索关键字
```

例如，创建一个 books 对象仓库，keyPath 为 isbn（书号）。代码如下所示：

```
var store = db.createObjectStore("books", {keyPath:"isbn"});
```

2）deleteObjectStore()方法删除对象仓库

```
db.deleteObjectStore(objectStoreName);      //基本语法
db.deleteObjectStore("books");              //举例-删除books对象仓库
```

3）createIndex()方法为对象仓库创建索引

```
var indexName=store.createIndex(index_name, index_key, {unique:
true|false});
```

代码中参数 index_name 是索引名称，例如 by_title 表示按标题建立索引；index_key 是索引键值名称；{unique:true|false}是可选项，表示是否唯一。true：唯一，false：不唯一。

**【例 17-1-4】**为 books 对象仓库按标题(title)和作者(author)建立索引。代码如下所示：

```
var titleIndex = store.createIndex("by_title", "title", {unique:false});
                                                              //标题索引
var authorIndex = store.createIndex("by_author", "author"); //作者索引
```

4）objectStoreNames 属性检查对象仓库是否存在

objectStoreNames 属性返回一个 DOMStringList 对象，里面包含了当前数据库所有“对象仓库”的名称。可以使用 DOMStringList 对象的 contains 方法，检查数据库是否包含某个“对象仓库”。代码格式如下：

```
if(!db.objectStoreNames.contains("books")) {    //判断某个对象仓库是否存在
    db.createObjectStore("books");              //不存在则创建该对象仓库
}
```

**4．使用事务**

需要使用事务在对象存储上执行所有读取和写入操作。类似于关系型数据库中事务的工作原理，IndexedDB 事务提供了数据库写入操作的一个原子集合，这个集合要么完全提

交，要么完全不提交。IndexedDB 事务还拥有数据库操作的一个中止和提交工具。

1）IndexedDB 中的事务模式

- readonly：提供对某个对象存储的只读访问，在查询对象存储时使用。
- readwrite：提供对某个对象存储的读取和写入访问权。
- versionchange：提供读取和写入访问权来修改对象存储定义，或者创建一个新的对象存储。

默认的事务模式为 readonly。可在任何给定时刻打开多个并发的 readonly 事务，但只能打开一个 readwrite 事务。因此，只有在数据更新时才考虑使用 readwrite 事务。单独的（表示不能打开任何其他并发事务）versionchange 事务操作一个数据库或对象存储。可以在 onupgradeneeded 事件处理函数中使用 versionchange 事务创建、修改或删除一个对象存储，或者将一个索引添加到对象存储。

2）创建事务的基本语法

```
var transaction = db.transaction(storeName, [transactionmode]);//基本语法
var objectStore =transaction.objectStore(storeName);  //获取指定的对象仓库
```

其中 db 为已连接的数据库对象；storeName 为对象仓库列表，列表是由多个对象仓库组成的字符串数组，不同的对象仓库名之间用逗号分隔。例如["students", "teachers"]表示同时为两个对象仓库创建一个事务。[transactionmode]为可选项，取值可以为 readonly、readwrite、versionchange。不设置此参数默认为只读。transaction()方法返回一个事务对象，该对象的 objectStore()方法用于获取指定的对象仓库。

**【例 17-1-5】**为 books、press 对象仓库创建一个读写事务。代码如下所示：

```
var transaction1= db.transaction("books", "readwrite");
                                   //为一个对象创建一个读写事务
var transaction2= db.transaction(["books","press"],"readwrite");
                                   //为两个对象库创建一个读写事务
var objectStore=transaction1.objectStore("books ");  //获取books对象仓库
```

3）transaction()方法的事件类型

该方法有三种事件，分别是中断、完成和错误。

- abort：事务中断。
- complete：事务完成。
- error：事务出错。

事件处理代码结构如下所示：

```
var transaction = db.transaction(["books"], "readonly");
transaction.oncomplete = function(event) {
  console.log("数据添加成功！");          //alert("数据保存成功！");
};
transaction.onerror = function(event) {
  console.log("Error",e.target.error.name);
  //错误处理
};
transaction.onabort = function(event) {
  alert("数据保存失败！");
};
```

**5. 数据库的增删改查**

数据库的增加、更新、删除和读取都会触发两个事件，分别是 success(检索请求成功)和 error(检索请求失败)，所以编程时需要为它们指派事件处理函数。

1）存储数据准备

给 books 对象仓库定义三个对象存放在 booklists 数组中。其中每对{}中定义一个对象，每个对象之间用逗号分隔，准备写入到对象仓库中。代码如下所示：

```
var booklists=[{title:"Web前端开发技术-Html、Css、JavaScript",author: "储久良",isbn: "9787302431695"},{title:"计算机组成原理(修订版)", author:"张功萱",
isbn: "9787302433637"},{title:"Java 2实用教程(第5版)", author: "耿祥义",
isbn:"9787302464259 "}];
```

2）数据库的增加、更新、删除

```
objectStore.add(objectName);     //添加数据，当关键字存在时数据不会添加
objectStore.put(objectName);
                          //更新数据，当关键字存在时覆盖数据，不存在时会添加数据
objectStore.delete(value);       //删除数据，删除指定的关键字对应的数据
objectStore.clear();             //清除objectStore
```

3）数据库的数据读取

```
var request = objectStore.get(value);   //查找数据，根据关键字查找指定的数据
```

- 常用方式。分配事件句柄，并绑定事件处理函数。

```
request.onsuccess=function(e){
   var books=e.target.result;
   console.log(books.title);            //控制台输出图书的标题
};
request.onerror=function(e){
console.log("数据读取失败！");          //控制台输出图书的标题
};
```

- 事件监听方式。分配事件句柄，并绑定事件处理函数。

```
request.addEventListener('success', function(event){    //增加事件监听器
   //异步查找后的调用函数，省略
}, false);
request.addEventListener('error', function(event){      //增加事件监听器
   //错误处理函数，省略
}, false);
```

**【例 17-1-6】**将已定义的三个对象添加到 books 对象仓库中。代码如下所示：

```
for(var i=0 ; i<booklists.length;i++){//采用for循环将三个对象添加到指定对象仓库
    request=objectStore.add(booklists[i]);
                                //添加对象到books中，也可以用put()方法
request.onerror = function(){
        console.error('数据库中已有该对象,不能重复添加!! ');
    };
request.onsuccess=function(){         //控制台输出或某个HTML标记内输出
        console.log('对象已成功存入对象仓库中！')
```

```
    };
}
```

**6．遍历数据 openCursor()方法**

使用对象仓库的 openCursor()方法可以实现遍历数据。该方法可以获取游标对象，然后利用游标移动来实现数据遍历。openCursor()方法还可以接受第二个参数，表示遍历方向，默认值为 next，其他值为 prev、nextunique 和 prevunique。后两个值表示如果遇到重复值，会自动跳过。openCursor()方法是异步执行的，有两个事件分别是 success（检索请求成功）和 error（检索请求失败），可以为它们指派事件处理函数。

1）非索引查找

**【例 17-1-7】**不使用索引，直接使用游标遍历数据。代码如下所示：

```
var tx=db.transaction(["books"], "readonly");   //创建事件对象
var objectStore=tx.objectStore("books");    //利用事务对象获取指定的对象仓库
var cursor=objectStore.openCursor();        //通过对象仓库打开游标
cursor.onsuccess=function(e){
    var result=e.target.result;             //获取结果集
    if(result){                     //条件成立时，控制台输出或某个HTML标记内输出信息
        console.log("key", result.key);     //输出键名，如isbn
        console.dir("data", result.value);  //列出该对象所有属性和方法
        result.continue();                  //游标移到下一个数据对象上
    }else{
 console.log("没有数据可遍历！");
      }
};
cursor.onerror=function(e){
console.log("没有数据可遍历！");
  };
```

代码中事件处理函数的参数为事件对象，该对象的 target.result 属性指向当前数据对象。当前数据对象的 key 和 value 分别返回键名和键值（即实际存入的数据）。continue()方法将光标移到下一个数据对象，如果当前数据对象已经是最后一个数据了，则光标指向 null。

编程时可以采用 console.log()方法来取代 alert()或 document.write()方法。采用 console.dir()可以显示一个对象所有的属性和方法。

2）IDBKeyRange 对象

通过索引可以读取指定范围内的数据。使用浏览器原生的 IDBKeyRange 对象能够生成指定范围的 range 对象。生成方法有四种。

- lowerBound()方法：指定范围的下限。
- upperBound()方法：指定范围的上限。
- bound()方法：指定范围的上下限。
- only()方法：指定范围中只有一个值。

**【例 17-1-8】**使用 IDBKeyRange 对象生成 range 对象的各种情形。代码如下所示：

```
var range1=IDBKeyRange.upperBound(x);           //All keys ≤ x
var range2=IDBKeyRange.upperBound(x, true);     //All keys < x
var range3=IDBKeyRange.lowerBound(y);           //All keys ≥ y
```

```
var range4=IDBKeyRange.lowerBound(y, true);          // All keys > y
var range5=IDBKeyRange.bound(x, y);                  // x≤ All keys ≤ y
var range6=IDBKeyRange.bound(x, y, true, true);      // x <All keys < y
var range7=IDBKeyRange.bound(x, y, true, false);     // x< All keys ≤ y
var range8=IDBKeyRange.bound(x, y, false, true);     // x ≤All keys < y
var range9=IDBKeyRange.only(z);                      // The key = z
```

代码中参数为 true 表示某个界限值为开区间；false 表示某个界限值为闭区间。如果有两个参数为逻辑值，分别表示下限值、上限值为开区间或闭区间。

3）按索引查找数据

使用对象仓库的 index()方法来实现检索。代码如下所示：

```
var index=objectStore.index(indexName);//indexName为已建立的索引名称
var cursor=index.openCursor(range);     //用IDBKeyRange生成范围range
cursor.addEventListener('success', function(event){      //启动成功监听
    var result = event.target.result;   //返回检索结果集
    if(result){
        console.log(result.value);      //输出数据
        result.continue();              //迭代，游标下移
    }
}, false);
cursor.addEventListener('error', function(event){console.log("失败！");
},false);
```

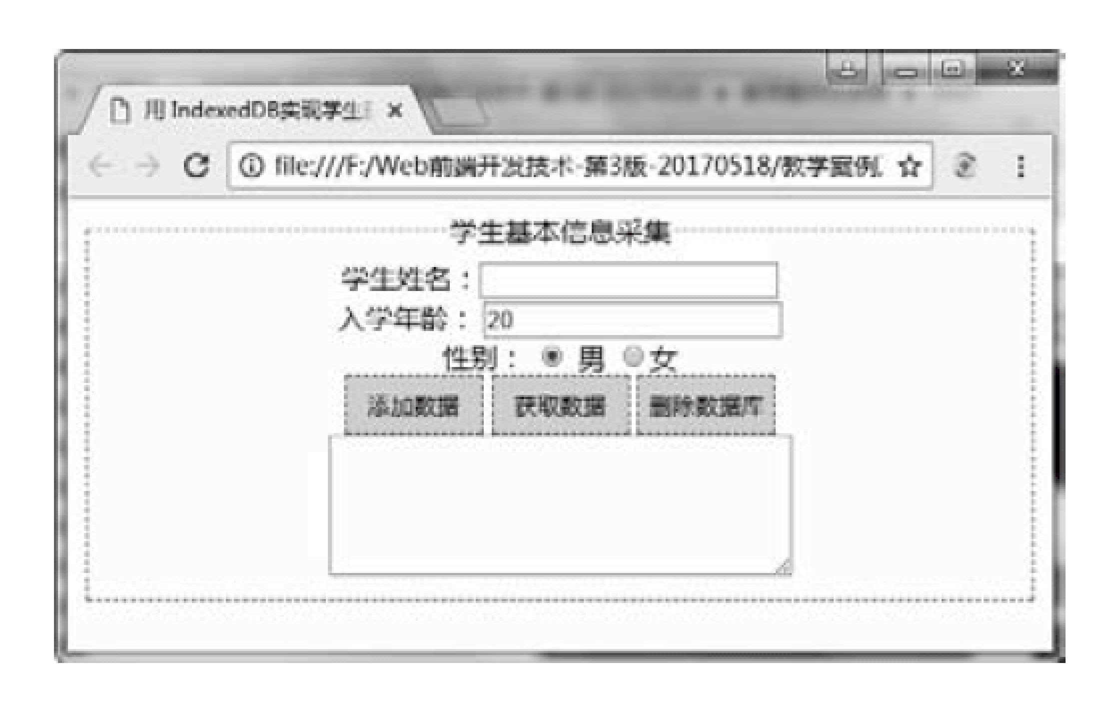
按索引的范围查找数据时需要定义 range 范围。当 range 为 null 时，查找所有数据；当 range 为指定值时，查找索引满足该条件对应的数据；当 range 为 IDBKeyRange 对象时，根据条件查找满足条件的指定范围的数据。

**【例 17-1-9】**按姓名的开始字母范围(B～D)来检索数据。部分代码如下所示：

```
var person = {                                          //定义对象
    name:name,
    email:email,
    created:new Date()
}
var transaction=db.transaction(["people"],"readonly");
                                                        //为people定义只读事件
var objectStore= transaction.objectStore("people");//获取people对象仓库
var index= objectStore.index("name");                   //按姓名进行索引
var range=IDBKeyRange.bound('B', 'D');                  //生成范围range对象
index.openCursor(range).onsuccess=function(e){
                                                //打开索引游标，启动成功监听事件
     var cursor=e.target.result;                        //获取结果集
     if(cursor){                                        //通过控制台输出相关信息
         console.log(cursor.key + ":");
         for(var field in cursor.value) {
             console.log(cursor.value[field]);
         }
         cursor.continue();                             //下移游标，迭代执行
    }
};
```

**【例 17-1-10】**使用 IndexedDB 实现学生基本信息采集系统。代码如下所示，页面效果如图 17-4 所示。学生信息包括学生姓名、入学年龄、性别等信息。根据如图 17-4 所示的

页面效果编写代码实现添加数据、获取数据和删除数据库等功能。



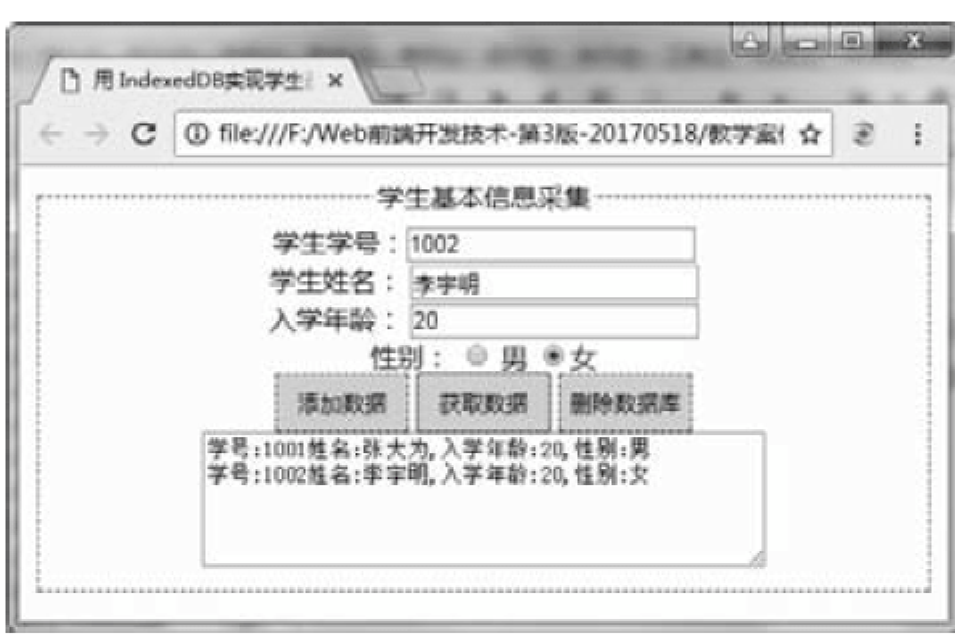




图 17-4　学生基本信息采集初始页面和添加与查询页面

```
<!-- edu_17_1_10.html -->
<!DOCTYPE html>
<html>
    <head>
        <meta charset="utf-8" />
        <title>用 IndexedDB实现学生基本信息采集</title>
        <style>
            /* 2.页面表现设计   */
            fieldset {text-align: center; border: 1px dashed #FF0000;}
            .myBtn {width: 80px;height: 35px;border: 1px dashed #0066FF;}
        </style>
        <script type="text/javascript">
            //定义全局变量结果集、IDBOpenDBRequest、对象仓库
            var db, request, objectStore;
            function createDB(dbName) {
                request = indexedDB.open(dbName, 1);
                request.onerror = function() {
                    alert("打开数据库失败:" + event.target.errorCode);
                }
                request.onsuccess = function(event) {
                    alert("打开数据库成功!");
                    db = event.target.result;
                    var transaction = db.transaction("user",
                    "readwrite");
                    objectStore = transaction.objectStore("user");
                }
                request.onupgradeneeded = function(event) {
                    alert("版本已经发生改变! ");
                    db = event.target.result; //获取结果集
                    objectStore = db.createObjectStore("user", {
                        keyPath: "userNo"
                    });
                    indexNo = objectStore.createIndex("by_userNo",
                    "userNo", {
                        unique: true
                    });
                }
            }
            createDB('userinfo'); //打开数据库
            function deleteDB(dbName) { //删除数据库
                request = indexedDB.deleteDatabase(dbName);
                request.onerror = function() {
```

```
                alert("删除数据库失败!");
            }
            request.onsuccess = function(event) {
                alert("删除数据库成功!");
                var ta = document.getElementById("display");
                ta.innerHTML = '';
                window.location.reload();
            }
        }
        function getObject() {
            var txtAear = document.getElementById("display");
            txtAear.innerHTML = ""; //获取数据前先清理一下页面已显示的数据
            if(!db) {
                alert("请先打开数据库! ");
                return false;
            }
            var store = db.transaction("user").objectStore("user");
            var keyRange = IDBKeyRange.lowerBound(0); //规定从0开始
            var cursorRequest = store.openCursor(keyRange);
                                                         //设置开启游标
            cursorRequest.onsuccess = function(e) {
                var result = e.target.result;
                if(!!result == false)
                    return;
                getOneObject(result.value);              //取一个对象数据
                result.continue();                       //这里执行轮询读取
            };
            cursorRequest.onerror = function(e) {
                alert("数据检索失败! ");
            };
        }
        function getOneObject(e) { //取一个对象数据
            var ta = document.getElementById("display");
            ta.innerHTML += "学号:" + e.userNo + "姓名:" + e.username
            + ",入学年龄:" + e.userage + ",性别:" + e.usersex + "\n";
        }
        function addObject() {
            var userID = document.getElementById("xuehao").value;
            var name = document.getElementById("name").value;
            var age = document.getElementById("age").value;
            var sex, flag = document.getElementById("nan").checked;
            sex = (flag) ? "男" : "女"; //条件表达式
      if(userID.length > 0 && name.length > 0 && age >= 0 && sex.length
      > 0) {
                //定义存储在对象仓库中的对象
                var detail = {
                    userNo: userID,username: name,userage:
                    age,usersex: sex
                }
                if(!db) {
                    alert("请先打开数据库! ");
                    return false;
                }
                var transaction = db.transaction(["user"],
                "readwrite");
                objectStore = transaction.objectStore("user");
                objectStore.add(detail);
```

```
                    alert("数据添加成功！");
                    myFrm.Reset();
                } else {
                    alert("输入数据不合法！请重新输入");
                    myFrm.xuehao.focus();
                }
            }
        </script>
    </head>
    <body>
        <!-- 1.页面内容设计   -->
        <form name="myFrm">
            <fieldset>
             <legend align="center">学生基本信息采集</legend>
             学生学号：<input type="text" name="xuehao" id="xuehao"
             required="required"><br /> 学生姓名：
             <input type="text" name="name" id="name" required=
             "required"><br />入学年龄：
             <input type="number" id="age" value="20" min=1><br /> 性别：
             <input type="radio" id="nan"name="xb" value="male" checked> 男
        <input type="radio" id="nv" name="xb" value="female">女<br />
        <input type="button" class="myBtn" onclick="addObject()"
        value="添加数据">
        <input type="button" class="myBtn" onclick="getObject()" value=
        "获取数据">
            <input type="button" class="myBtn" onclick="deleteDB
           ('userinfo')"value="删除数据库"><br/>
             <textarea id="display" rows="5" cols="45"> </textarea>
            </fieldset>
        </form>
    </body>
</html>
```

代码解释：案例中创建的数据库名为 userinfo，创建对象仓库名为 user。createDB (dbName) 函数的功能是根据指定参数创建数据库。deleteDB(dbName) 函数的功能是根据指定参数删除数据库。getObject()函数的功能是获取满足条件的所有对象，并在多行文本域中分行显示。getOneObject(e)函数的功能是读取某一个对象，并显示在多行文本域中。addObject()函数的功能是将用户在表单中输入的信息添加到对象仓库 user 中。

# 17.2　HTML5 Canvas 画布

HTML5 的 canvas 标记用于图形的绘制，并通过脚本 (JavaScript)来完成绘图。canvas 标记本身并没有绘图能力，所有的绘制工作必须在 JavaScript 内部完成。canvas 标记作为图形的容器，可以通过多种方法使用 Canvas 绘制路径、盒、圆、字符以及添加图像。

## 17.2.1　canvas 标记

canvas 标记是双标记，必须设置宽度、高度及 id。

**1．基本语法**

```
<canvas id="oneCanvas" width="" height=""></canvas>
```

默认情况下 canvas 标记的 width 为 300px、高度 height 为 200px，页面上没有边框和内容。除非通过 CSS 定义边框样式，才可以显示效果，所以必须借助于 JavaScript 才能绘图。常用的绘制颜色、样式和阴影的属性及说明如表 17-1 所示。绘制一个矩形代码如下：

**表 17-1　颜色、样式和阴影的属性及说明**

| 属　　性 | 说　　明 |
|---|---|
| fillStyle | 设置或返回用于填充绘画的颜色、渐变或模式 |
| strokeStyle | 设置或返回用于笔触的颜色、渐变或模式 |
| shadowColor | 设置或返回用于阴影的颜色 |
| shadowBlur | 设置或返回用于阴影的模糊级别 |
| shadowOffsetX | 设置或返回阴影与形状的水平距离 |
| shadowOffsetY | 设置或返回阴影与形状的垂直距离 |

```
<body>
    <canvas id="aCanvas" width="200" height="100"></canvas>
    <script type="text/javascript">
        var myCanvas=document.getElementById("aCanvas");
                                            //获取Canvas对象
        var conText=myCanvas.getContext("2d");//获取绘图环境（上下文环境）
        conText.fillStyle="#FF0000";        //设置填充样式
        conText.fillRect(10,10,150,75);     //绘制矩形
    </script>
</body>
```

**2．绘制图形**

利用 canvas 标记绘制图形一般需要经过下列步骤：

（1）在 body 标记中插入 canvas 标记，并设置 id、width、height。

```
<canvas id="aCanvas" width="200" height="100"></canvas>
```

（2）在 body 标记中插入 script 标记，并在该标记内插入相关 JavaScript 代码。

（3）通过 id 获取页面上 canvas 对象。

```
var myCanvas=document.getElementById("aCanvas");   //获取Canvas对象
```

（4）创建具有绘图功能的环境对象 context，参数为 2d 或 3d。

```
var conText=myCanvas.getContext("2d");      //获取绘图环境（也称上下文环境）
```

（5）在绘图环境对象内绘图。

- 填充。分为填充样式和填充图形。

```
conText.fillStyle="#FF0000";                //设置填充样式
conText.fillRect(10,10,150,75);             //绘制矩形
```

- 绘制边框（外轮廓）。绘制样式和绘制图形及绘制线条的宽度（相当于画笔粗细）。

```
conText.strokeStyle="#FF0000";              //设置边框样式
conText.lineWidth=8;                        //图形边框宽度，不加单位px
```

```
conText.strokeRect(0,0,200,100);          //绘制边框
```

- 清除矩形区域

```
conText.clearRect(x,y,width,height)
```

**【例 17-2-1】** 用 canvas 标记绘制矩形。代码如下，页面效果如图 17-5 所示。

视频讲解

```
<!-- edu_17_2_1.html -->
<!doctype html>
<html lang="en">
   <head>
      <meta charset="UTF-8">
      <title>Canvas绘制矩形</title>
   </head>
   <body>
      <canvas id="oneCanvas" width="" height="" style="border:1px solid
      blue"></canvas>
      <script type="text/javascript">
         var myCanvas=document.getElementById("oneCanvas");
         //获取画布对象
         var conText=myCanvas.getContext("2d");
         //获取绘图环境（上下文环境）
         conText.fillStyle="#00FF00";          //设置填充样式
         conText.fillRect(0,0,200,100);        //填充矩形
         conText.strokeStyle="#FF0000";        //设置边框样式
         conText.lineWidth=8;                  //图形边框宽度
         conText.strokeRect(0,0,200,100);      //绘制边框
      </script>
   </body>
</html>
```




图 17-5　Canvas 绘制矩形效果图

## 17.2.2　Canvas 坐标

Canvas 画布是一个二维网格，分 X 轴和 Y 轴，其中 X 轴方向从左向右，Y 轴方向从上到下。绘制矩形方法是 fillRect (X,Y,width,height)，其中 X、Y 参数分别表示 X 轴、Y 轴的坐标，其余两个参数分别表示矩形的宽度和高度，如图 17-6 所示。Canvas 绘图环境对象的左上角坐标为(0，0)。例如 fillRect (0,0,200,175)表示画布上绘制 200×175px 的矩形，从左上角(0，0)开始绘制。

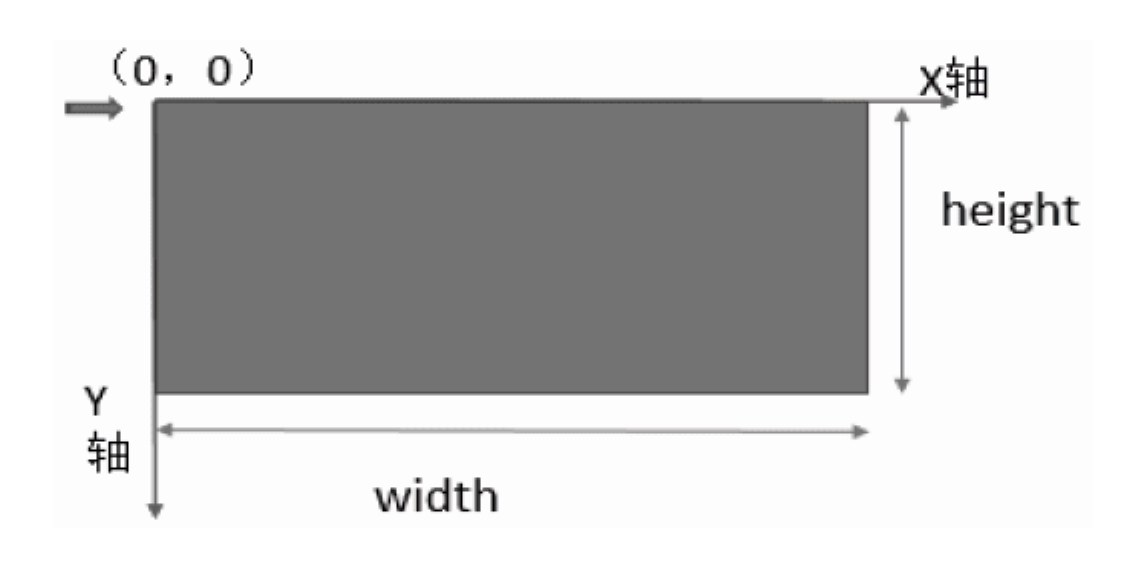



图 17-6　Canvas 坐标系统

### 17.2.3　Canvas 路径

在 Canvas 上除了绘制矩形、正方形和直线外，需要使用路径来进行绘图。绘制前需要使用 beginPath()方法开始路径，然后形成绘制路径，结束后需要使用 closePath()方法关闭路径。最后才开始填充或绘制。常用绘制路径方法及说明如表 17-2 所示。

表 17-2　绘制路径常用方法及说明

| 方　法 | 说　明 |
|---|---|
| fill() | 填充当前绘图（路径） |
| stroke() | 绘制已定义的路径（边框） |
| beginPath() | 起始一条路径，或重置当前路径 |
| moveTo() | 把路径移动到画布中的指定点，不创建线条 |
| closePath() | 创建从当前点回到起始点的路径 |
| lineTo() | 添加一个新点，然后在画布中创建从该点到最后指定点的线条 |
| clip() | 从原始画布剪切任意形状和尺寸的区域 |
| quadraticCurveTo() | 创建二次贝塞尔曲线 |
| bezierCurveTo() | 创建三次贝塞尔曲线 |
| arc() | 创建弧/曲线（用于创建圆形或部分圆） |
| arcTo() | 创建两切线之间的弧/曲线 |
| isPointInPath() | 如果指定的点位于当前路径中，则返回 true，否则返回 false |

下面以在 Canvas 中绘制圆形为例，说明具体步骤。

**1．基本语法**

```
context.arc(x, y, radius, startAngle,endAngle, anticlockwise)
```

**2．语法说明**

其中 x、y、radius 分别圆心 x 坐标、y 坐标和半径。startAngle、endAngle 分别表示绘制开始角度和绘制结束角度。anticlockwise 表示绘制方向是否是逆时针的。true 为逆时针，false 为顺时针。

**3．具体步骤**

前四步与绘制矩形步骤相同，此处省略。后续步骤如下：

（1）开始路径。方法为 conText.beginPath()。

（2）绘制路径。方法如下：

```
conText.arc(150, 150, 100, 0, Math.PI * 2, true);   //绘制路径
```

（3）关闭路径。方法为 conText. closePath()。

（4）绘图。分为填充和绘制圆形边框，与绘制矩形类似。

```
conText.fillStyle = 'rgba(255,0,0,0.75)';          //填充样式
conText.fill();                                     //填充绘图
conText.strokeStyle = 'rgba(0,255,0,0.50)';        //绘图样式
conText.lineWidth=20;                               //绘制边框宽度
conText.stroke();                                   //填充绘图
```

其中 rgba 是以 RGB 为基础再加上不透明度的属性。格式如下：

```
rgba(red, green, blue, opacity)
```

opacity 属性表示透明度。允许的值为 0～1 之间带有小数的数值。

**【例 17-2-2】**用 Canvas 标记绘制圆形。代码如下，页面效果如图 17-7 所示。

视频讲解

```
<!-- edu_17_2_2.html -->
<!doctype html>
<html lang="en">
 <head>
     <meta charset="UTF-8">
     <title>Canvas绘制圆形</title>
 </head>
 <body>
     <canvas id="oneCanvas" width="300" height="300" style=
     "background:#F0F0F0;"></canvas>
     <script type="text/javascript">
       var myCanvas=document.getElementById("oneCanvas");
                                         //获取Canvas对象
       var conText=myCanvas.getContext("2d");  //获取绘图环境（上下文环境）
       conText.beginPath();              //开始路径
       conText.arc(150, 150, 100, 0, Math.PI * 2, true);     //绘制路径
       conText.closePath();              //关闭路径
       conText.fillStyle = 'rgba(255,0,0,0.75)';
                                         //填充样式，第4个参数表示透明度
       conText.fill();                   //填充绘图
       conText.strokeStyle = 'rgba(0,255,0,0.50)';
                                         //绘图样式，第4个参数表示透明度
       conText.lineWidth=20;             //绘制边框宽度
       conText.stroke();                 //绘制边框
     </script>
 </body>
</html>
```

## 17.2.4 Canvas 绘制线段

利用 Canvas 标记可以绘制线段。常用的方法有 moveTo(x,y)和 lineTo(x,y)。

**1．基本语法**

```
context.moveTo(x,y)          //定义线段开始坐标，x为X轴坐标，y为Y轴坐标
context.lineTo(x,y)          //定义线段结束坐标，x为X轴坐标，y为Y轴坐标
```

**2．语法说明**

两个方法的参数相同，x 为 X 轴坐标，y 为 Y 轴坐标。moveTo()表示设置线段的起点；lineTo()表示设置线段的终点。第 1 条线段起点用 moveTo()方法与 lineTo()方法作用相同。从第 2 条线段开始，如果没有 moveTo()设置线段起点，则说明下一条线段的起点与上一条线段的终点相同。




视频讲解

【例 17-2-3】用 Canvas 标记绘制直线。代码如下，页面效果如图 17-8 所示。





图 17-7　Canvas 绘制圆形

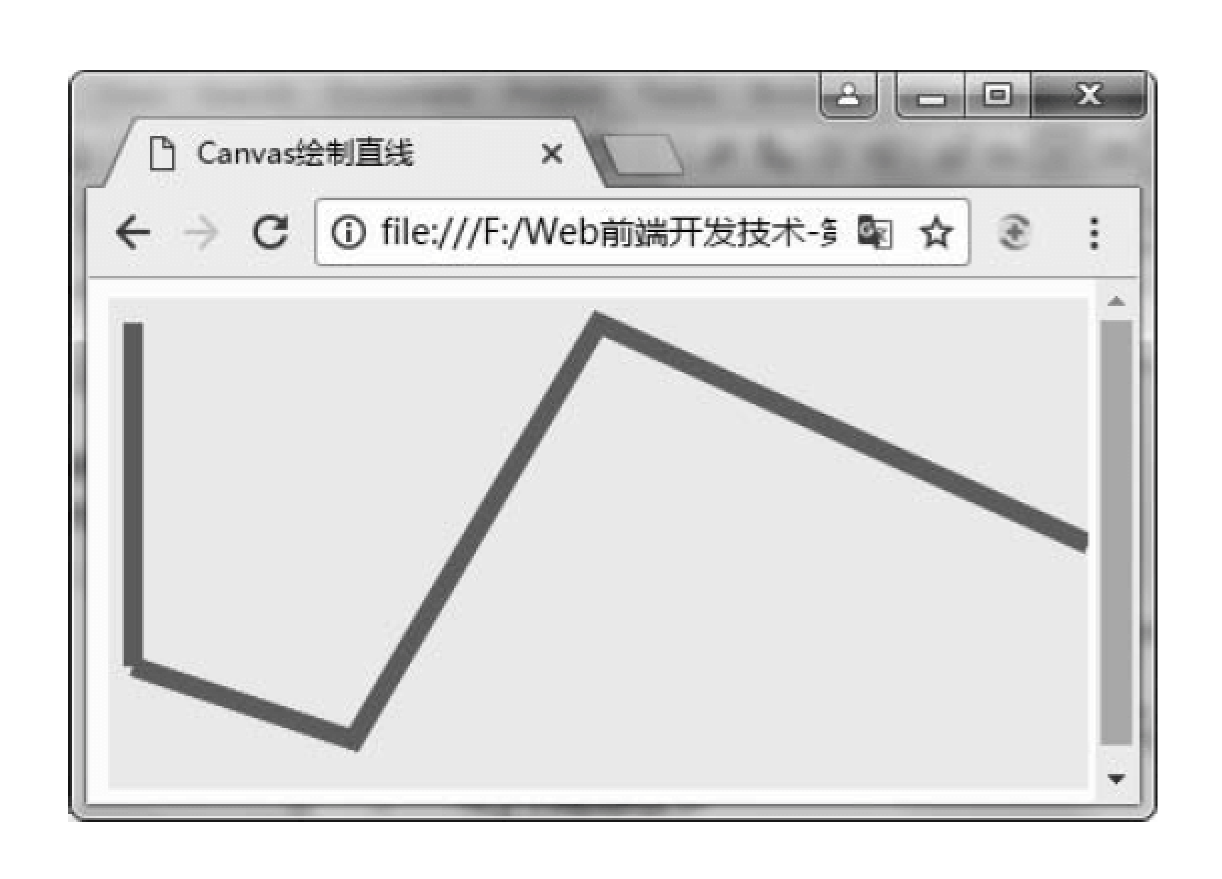



图 17-8　Canvas 绘制直线

```
<!-- edu_17_2_3.html -->
<!doctype html>
<html lang="en">
 <head>
  <meta charset="UTF-8">
  <title>Canvas绘制直线</title>
 </head>
 <body>
  <canvas id="oneCanvas" width="400" height="200" style="background:
  #F0F0F0;"></canvas>
  <script type="text/javascript">
    var myCanvas=document.getElementById("oneCanvas");//获取Canvas对象
    var conText=myCanvas.getContext("2d");  //获取绘图环境（上下文环境）
    conText.strokeStyle = "rgb(250,0,0)";
    conText.fillStyle = "rgb(250,0,0)"
    conText.moveTo(10,10);                  //第1条线起点
    conText.lineTo(10,150);                 //第1条线终点
    conText.moveTo(10,150);                 //第2条线起点
    conText.lineTo(100,180);                //第2条线终点
    conText.lineTo(200,10);                 //第3条以第2条的终点为起点-终点
    conText.lineTo(400,100);                //第4条以第3条的终点为起点-终点
    conText.lineWidth=8;                    //绘制边框宽度
    conText.stroke();                       //绘制边框
  </script>
 </body>
</html>
```

### 17.2.5　Canvas 绘制文本

利用 Canvas 除了可以绘制矩形、圆形等，还可以绘制文本。

**1．基本语法**

```
Context.fillText(text,x,y)                          //在canvas上绘制实心的文本
Context.strokeText(text,x,y)                        //在canvas上绘制空心的文本
context.font="font-style font-weight font-variant font-size/line-height
font-family"
context.textAlign="start|end|left|right|center"   //水平对齐
context.textBaseline="alphabetic|top|hanging|middle|ideographic|bottom";
                                                    //垂直对齐
```

**2．语法说明**

context.fillText(text,x,y)：填充文本。

context.strokeText(text,x,y)：绘制文本轮廓。其中参数 text 表示要绘制的文本；参数 x 表示文本起点的 X 坐标轴；y 表示文本起点的 Y 坐标轴。

context.font：设置字体样式。设置方法与 CSS 的 font 属性方法相同。

context.textAlign：设置或返回文本内容的当前对齐方式。其值可设置为 start（文本在指定的位置开始）、end（文本在指定的位置结束）、left（文本左对齐）、center（文本的中心被放置在指定的位置）、right（文本右对齐）。

context.textBaseline：设置或返回在绘制文本时使用的当前文本基线（垂直对齐方式）。其值可设置为 top（顶部）、hanging（悬挂，比 top 略高些）、middle（中部）、alphabetic（默认，普通的字母基线）、ideographic（表意基线，与 bottom 同效果）、bottom（底部）。textBaseline 属性在不同的浏览器上效果不同，特别是使用"hanging"或"ideographic"时，在不同浏览器中效果不同。绘制文本格式如下所示：

```
context.textAlign="start";                          //设置提示信息水平对齐方法
context.font="24px 黑体";                           //设置提示信息字体
context.fillText("文本基线位置：",0,220);           //设置提示信息
```

### 17.2.6　Canvas 渐变

渐变可以填充在矩形、圆形、线条、文本等。各种形状可以自己定义不同的颜色。

**1．基本语法**

```
var grad=context.createLinearGradient(xstart,ystart,xend,yend);
                                   //创建线条渐变
grad.addColorStop(offset,color);   //指定颜色停止,offset可以是0至1
var grad=context.createRadialGradient(xstart,ystart,radiusstart,xend,
yend,radiusend)
                                   //圆径向渐变
context.fillStyle=grad;            //渐变对象变量
context.fillRect(x,y,width,height);
```

**2．语法说明**

线条渐变 createLinearGradient()中参数 xstart 表示渐变开始点 x 坐标；ystart 表示渐变开始点 y 坐标；xEnd 表示渐变结束点 x 坐标；yEnd 表示渐变结束点 y 坐标。addColorStop()

中参数 offset 表示设定的颜色离渐变结束点的偏移量(0～1)；color 表示绘制时要使用的颜色。

createRadialGradient()中参数有 6 个，前 3 个参数表示径向渐变开始圆心坐标和半径；后 3 个参数表示径向渐变结束圆心坐标和半径。

当使用渐变对象时，必须使用两种或两种以上的停止颜色，设置 fillStyle 或 strokeStyle 的值为渐变，然后绘制形状，如矩形、文本或一条线。

【例 17-2-4】绘制渐变线条部分代码如下所示，效果如图 17-9 右中部所示。

```
var grad=context.createLinearGradient(50,280,400,50);  //创建线条渐变
grad.addColorStop(0,"#FF0000");                        //设置偏移量为0、渐变停止颜色
grad.addColorStop(1,"#00FF00");                        //设置偏移量为1、渐变停止颜色
context.fillStyle=grad;                                //设置填充样式为渐变
context.fillRect(50,280,400,50);                       //填充矩形
```




图 17-9　综合运用 Canvas 绘制文本、图像、渐变

## 17.2.7　Canvas 绘制图像

把一幅图像放置到画布上，即在 Canvas 上画出图像。

**1. 基本语法**

```
context.drawImage(image,x,y);                   //在坐标(x,y)处开始绘制图像image
context.createPattern(image,type);                      //图像平铺
context.clip()                                          //图像裁剪
var imagedata=context.getImageData(sx,sy,sw,sh);        //像素处理
context.drawImage(image,x,y,width,height)       //按指定宽度和高度绘图
context.drawImage(image,sx,sy,sw,sh,dx,dy,dw,dh);
                                                //选取图像的部分矩形区域进行绘制
```

**2. 语法说明**

drawImage(image,x,y)方法中参数说明如下：x 表示绘制图像的 x 坐标；y 表示绘制图

像的 y 坐标；image 表示图像对象。

createPattern(image,type)方法中参数说明：type 取值有 4 种，分别为 no-repeat 表示不平铺、repeat-x 表示横方向平铺、repeat-y 表示纵方向平铺、repeat 表示全方向平铺。image 为图像对象。

context.clip()方法只绘制封闭路径区域内的图像，不绘制路径外部图像。使用时先创建裁剪区域，再绘制图像。代码如下所示，效果如图 17-9 右边所示。

```
context.beginPath();                                //开始路径
context.arc(200, 150, 100, 0, Math.PI * 2, true);   //形成圆形路径
context.closePath();                                //结束路径
context.clip();                               //从原image中剪裁出圆形部分图像
context.drawImage(image,50,380);              //除剪裁部分，其余不可见
```

drawImage(image,x,y,width,height)方法中参数说明如下：x 表示绘制图像的 x 坐标；y 表示绘制图像的 y 坐标；width 表示绘制图像的宽度；height 表示绘制图像的高度。

drawImage(image,sx,sy,sw,sh,dx,dy,dw,dh)方法中参数说明如下：sx 表示图像上的 x 坐标；sy 表示图像上的 y 坐标；sw 表示矩形区域的宽度；sh 表示矩形区域的高度；dx 表示画在 canvas 的 x 坐标；dy 表示画在 canvas 的 y 坐标；dw 表示画出来的宽度；dh 表示画出来的高度。各个参数的标注如图 17-10 所示。

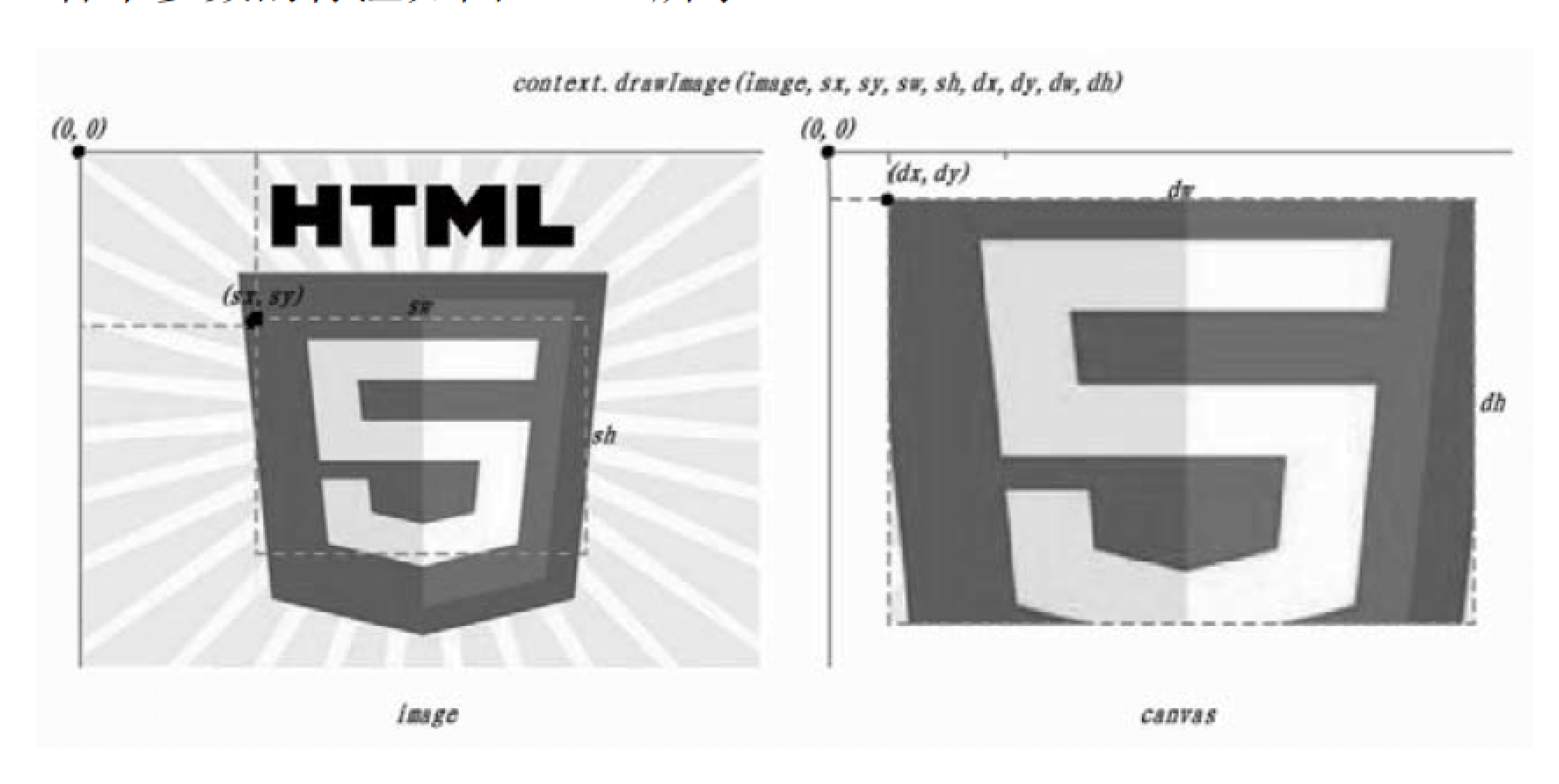



图 17-10　选取图像的部分矩形区域进行绘制

【例 17-2-4】综合运用 Canvas 绘制文本、图像、渐变。代码如下，页面效果如图 17-9 所示。

```
<!-- edu_17_2_4.html -->
<!doctype html>
<html lang="en">
    <head>
        <meta charset="UTF-8">
        <title>Canvas绘制文本、图像、渐变</title>
        <script type="text/javascript">
        <!--
            function showPage(){
            var myCanvas=document.getElementById("oneCanvas");
                                                    //获取画布对象
```

```
        var conText=myCanvas.getContext("2d");     //获取绘图环境对象
        conText.strokeStyle = "rgb(250,0,0)";
        conText.fillStyle = "rgb(0,0,0)"
        //在X轴150处绘制垂直红线
        conText.textAlign="start";           //设置提示信息水平对齐方法
        conText.font="24px 黑体";                  //设置提示信息字体
        conText.fillText("文本对齐方式：",0,24);     //设置提示信息
        conText.strokeStyle="red";
        conText.moveTo(350,20);
        conText.lineTo(350,170);
        conText.stroke();
        //绘制文本-textAlign属性应用
        conText.font="24px Arial";
        conText.textAlign="start";
        conText.fillText("在指定位置开始start",350,60);
        conText.textAlign="end";
        conText.fillText("在指定位置结束end",350,80);
        conText.textAlign="center";
        conText.fillText("文本中心在指定位置center",350,120);
        conText.lineWidth=1;
        conText.fill();
        //在Y轴250处画一条水平红线
        conText.textAlign="start";         //设置提示信息水平对齐方法
        conText.font="24px 黑体";                  //设置提示信息字体
        conText.fillText("文本基线位置：",0,220);    //设置提示信息
        conText.strokeStyle="red";
        conText.moveTo(0,250);
        conText.lineTo(700,250);
        conText.stroke();
        //每个在y = 250处设置不同的textBaseline值，显示单词的位置
        conText.font="20px Arial";
        conText.textBaseline="top";
        conText.fillText("Top-Hag",20,250);        //Hag表示字母组合
        conText.textBaseline="bottom";
        conText.fillText("Bottom-aXg",100,250);    //aXg表示字母组合
        conText.textBaseline="middle";
        conText.fillText("Middle",220,250);
        conText.textBaseline="alphabetic";
        conText.fillText("Alphabetic-aXg",300,250); //aXg表示字母组合
        conText.textBaseline="ideographic";
        conText.fillText("ideographic-aXg",460,250);
                                                   //aXg表示字母组合
        conText.textBaseline="hanging";
        conText.fillText("Hanging",620,250);
        //绘制渐变
        conText.font="20px 黑体";
        conText.textBaseline="bottom";
        conText.fillText("渐变：",0,320);
```

```
            var grad=conText.createLinearGradient(50,280,400,50);  //创建线条渐变
            grad.addColorStop(0,"#FF0000");          //设置渐变停止颜色1
            grad.addColorStop(1,"#00FF00");          //设置渐变停止颜色2
            conText.fillStyle=grad;                  //设置填充样式为渐变
            conText.fillRect(50,280,400,50);         //填充矩形
            /*绘制图像*/
            var myCanvas=document.getElementById("oneCanvas");  //获取画布对象
            var conText=myCanvas.getContext("2d");   //获取绘图环境对象
            conText.font="20px 黑体";
            conText.textBaseline="bottom";
            conText.fillText("图像：",0,380);
            var img=new Image();
            img.src="45567.jpg";
            conText.drawImage(img,50,380);        //在指定位置处开始绘图
            conText.drawImage(img,450,680,100,100);//按指定宽度和高度绘图
            /* 选取图像的部分矩形区域进行绘制 */
            conText.drawImage(img,200,200,100,100,550,660,120,120);
            /* 图像圆形剪裁 */
            conText.fillStyle="#F8F8F8";          //填充样式
            conText.fillRect(680,378,400,400);    //填充
            conText.beginPath();                  //开始路径
            conText.arc(890, 578, 100, 0, Math.PI * 2, true);
                                                  //形成圆形路径
            conText.closePath();                  //结束路径
            conText.clip();                       //圆形剪裁
            conText.drawImage(img,680,378);
                                        //按圆形剪裁图像，其余部分不可见
        }
        //-->
        </script>
    </head>
    <body onload="showPage();">
        <div>
            <img src="45567.jpg" id="myimg" style="float:left;"/>
            <canvas id="oneCanvas" width="1100" height="800" style=
            "background:#F0F0F0;"></canvas>
        </div>
    </body>
</html>
```

# 17.3　HTML5 拖放

拖放（Drag 和 Drop）是一种常见的特性，即抓取对象以后拖到另一个位置。拖放是 HTML5 标准的组成部分，任何元素都能够拖放，只要设置 draggable 属性为 true 即可。IE 9、Firefox、Opera、Chrome 以及 Safari6 等高版本的浏览器均支持拖放。

为了使元素可拖动，可采取下列步骤。

## 17.3.1　设置元素为可拖放

需要将该元素的 draggable 属性值设置为 true。代码如下：

```
<img id="" src="" draggable="true" >
```

### 17.3.2 拖放事件

拖动过程会触发很多事件，事件、事件属性及说明如表 17-3 所示。

表 17-3 拖放过程发生的事件及说明

| 事 件 | 事件属性 | 说 明 |
|---|---|---|
| dragstart | ondragstart | 网页元素开始拖动时触发 |
| drag | ondrag | 被拖动的元素在拖动过程中持续触发 |
| dragenter | ondragenter | 被拖动的元素进入目标元素时触发，应在目标元素监听该事件 |
| dragleave | ondragleave | 被拖动的元素离开目标元素时触发，应在目标元素监听该事件 |
| dragover | ondragover | 被拖动元素停留在目标元素之中时持续触发，应在目标元素监听该事件 |
| drop | ondrop | 拖动操作结束，放置元素时触发。监听器负责检索被拖动的数据以及在放置位置插入它 |
| dragend | ondragend | 网页元素拖动结束时触发 |

### 17.3.3 dataTransfer 对象

dataTransfer 对象，它是事件 event 对象的一个属性，用于从被拖动元素向放置目标传递字符串格式的数据，可在拖放事件的事件处理程序中访问 dataTransfer 对象。该对象的常用属性及说明如表 17-4 所示，常用方法及说明如表 17-5 所示。

表 17-4 dataTransfer 对象常用的属性及说明

| 属 性 | 说 明 |
|---|---|
| dropEffect | 拖放的操作类型，决定了浏览器如何显示鼠标形状，其值可为 copy、move、link 和 none |
| effectAllowed | 指定所允许的操作，其值可为 copy、move、link、copyLink、copyMove、linkMove、all、none 和 uninitialized（默认值，等同于 all，即允许一切操作） |
| files | 包含一个 FileList 对象，表示拖放所涉及的文件，主要用于处理从文件系统拖入浏览器的文件 |
| types | 存储在 dataTransfer 对象的数据的类型 |

例如，在拖动开始时，在 dataTransfer 对象上存储一条文本信息，内容为“Hello World！”。当拖放结束时，可以用 getData()方法取出这条信息。代码如下所示：

```
draggableElement.addEventListener('dragstart', function(event) {
    event.dataTransfer.setData('text', 'Hello World!');      //存储信息
});
```

表 17-5 dataTransfer 对象常用的方法及说明

| 属 性 | 说 明 |
|---|---|
| setData(format, data) | 在 dataTransfer 对象上存储数据。第一个参数 format 用来指定存储的数据类型，例如 text、url、text/html 等 |
| getData(format) | 从 dataTransfer 对象取出数据 |
| clearData(format) | 清除 dataTransfer 对象所存储的数据。如果指定了 format 参数，则只清除该格式的数据，否则清除所有数据 |

续表

| 属　　性 | 说　　明 |
| --- | --- |
| setDragImage(imgElement, x, y) | 指定拖动过程中显示的图像。默认情况下，许多浏览器显示一个被拖动元素的半透明版本。参数 imgElement 必须是一个图像元素，而不是指向图像的路径，参数 x 和 y 表示图像相对于鼠标的位置 |

### 17.3.4　拖放操作实现步骤

拖放元素的过程可分为创建可拖放对象、设置放置对象两个步骤。具体如下：

**1．创建一个可拖放对象**

以一个<img>标记为例，要使该标记成为拖放对象，需要设置该元素的 draggable 属性为 true。同时给该标记的 dragstart 事件设置一个事件监听器存储拖放数据。事件监听器 dragstart 会设置允许的效果(copy、move、link 或者是组合形式的)。ondragstart 属性绑定 drag(event)函数，它规定了被拖动的数据。通过 dataTransfer.setData()方法设置被拖放数据的数据类型和值。具体代码格式如下：

```
<img src="45567.jpg" draggable="true" ondragstart="drag(event)" id="drag1" />
function drag(ev) {ev.dataTransfer.setData("Text",ev.target.id); }
```

参数 1 是数据类型，值为"Text"；参数 2 是数据信息，值为可拖动元素的 id ("drag1")。

**2．设置放置对象**

能够接受拖放元素的对象称为放置对象（或目标对象），放置对象至少要监听两个事件。

（1）dragover 事件。该事件对应的事件句柄 ondragover，被拖动元素停留在放置对象之中时持续触发。默认方式下，无法将数据/元素放置到其他元素中，需要设置允许放置，必须阻止对元素的默认处理方式。通过给 ondragover 事件属性绑定 allowDrag(event)函数，在函数中使用 event.preventDefault()方法来实现阻止默认处理方式。代码如下所示：

```
<div id="div1" ondrop="drop(event)" ondragover="allowDrop(event)">
function allowDrop(ev){
    ev.preventDefault();              //阻止对元素的默认处理方式
}
```

（2）drop 事件。该事件对应的事件句柄 ondrop，允许执行真正的放置。ondrop 属性绑定 drop(event)函数完成放置功能。当放置被拖数据时，会发生 drop 事件。该事件将阻止对元素的默认处理方式、获得拖放元素的数据信息、添加被拖放的元素。代码如下所示：

```
function drop(ev){            //放置
  ev.preventDefault();        //阻止对元素的默认处理方式，默认行为是以链接形式打开
  var data=ev.dataTransfer.getData("Text");//获取拖放数据，data中存储的元素id
  ev.target.appendChild(document.getElementById(data));
                              //被拖元素追加到放置元素中
}
```

以图层 div 为例，把 div 作为目标对象，设置图层的 ondrop 和 ondragover 事件属性，并绑定相关事件处理代码。代码如下所示：

```
<div id="div1" ondrop="drop(event)" ondragover="allowDrop(event)">
```

【例 17-3-1】HTML5 拖放图像应用。代码如下，页面效果如图 17-11 所示。







图 17-11　HTML5 拖放图像应用

视频讲解

```
<!-- edu_17_3_1.html -->
<!doctype html>
<html lang="en">
    <head>
        <meta charset="UTF-8">
        <title>HTML5拖放图像</title>
        <style type="text/css">
            #div1, #div2 {float:left; width:200px; height:200px;
              margin:15px;padding:15px;border:1px dashed #0066ff;}
            #drag1{width:200px;height:200px;}
        </style>
        <script type="text/javascript">
            function $(id){return document.getElementById(id);}//获取元素
            function allowDrop(ev){
                //阻止对元素的默认处理方式
                ev.preventDefault();}
            function drag(ev){
                //设置被拖数据的数据类型和值
               ev.dataTransfer.setData("Text",ev.target.id);}
            function drop(ev){
                ev.preventDefault();//阻止对元素的默认处理方式
                var data=ev.dataTransfer.getData("Text");//获得被拖的数据
                ev.target.appendChild($(data));//添加拖放元素
            }
        </script>
    </head>
    <body>
    <h3>两个div中图像互相拖放</h3><hr>
    <div id="div1" ondrop="drop(event)" ondragover="allowDrop(event)">
    <img src="45567.jpg" draggable="true" ondragstart="drag(event)"
    id="drag1"/>
    </div>
    <div id="div2" ondrop="drop(event)" ondragover="allowDrop(event)">
    </div>
    </body>
</html>
```

# 17.4　HTML5 Web Worker

在 HTML5 中提出了 Web Worker（工作线程）的概念，并且规范出 Web Worker 的三大主要特征：能够长时间运行（响应），理想的启动性能以及理想的内存消耗。Web Worker 允许开发人员编写能够长时间运行而不被用户所中断的后台程序，去执行事务或者逻辑，并同时保证页面对用户的及时响应。

## 17.4.1　Web Worker 的工作原理

Web Worker 的工作原理是在包含 JavaScript 脚本的 Web 页面中，运行的 JS 脚本称为主线程，并在主线程中使用 Worker 类创建一个 Worker，并向其传入一个参数，该参数是需要在另一个线程中运行的 JavaScript 文件名称 (myWorker.js)，然后在这个实例上监听 onmessage 事件。利用这个 JavaScript 文件来运行一个新的线程，起到互不阻塞执行的效果。主线程和新线程之间数据交换可通过 postMessage()方法和通过 onmessage 捕获来传递和接收数据。

## 17.4.2　创建 Web Worker 文件

利用 JavaScript 创建一个外部 Web Worker 文件 myWorker.js。它是一个独立的 JavaScript 脚本文件。主要功能是每隔 1s 随机产生 10 个 100 以内的 2 位整数，并保存在数组中，并通过 postMessage() 方法将数组元素传递给主线程。其调用方法如下：

```
worker.postMessage(data)                //data可以是一个字符串或者JSON对象
```

编写外部 JavaScript 文件，已经详细注解了代码的功能。代码如下所示：

```
/*  myWorker.js，每隔1s随机产生10个10～99之间的整数 */
var tenIntger=new Array();                  //定义保存随机2位整数的数组
function createTenIntger(){                 //产生10个随机整数
   for (var j=0;j<10;j++)                   //循环10次
   {    //利用数学函数随机产生10~99之间的整数，并存入数组中
        tenIntger[j]=Math.floor(Math.random()*90+10);
  }
   postMessage(tenIntger.sort());           //数组元素排序后传递给主线程
   setTimeout("createTenIntger()",1000);    //每隔1s重新产生1次
}
createTenIntger();                          //调用方法
```

通常情况下，Web Worker 不用于如此简单的脚本，而是用于更耗费 CPU 资源的任务。

## 17.4.3　创建 Web Worker 对象

编辑完成 Web Worker 文件后，需要利用 Worker 类创建一个新的 Worker 线程，并为其传入一个参数，该参数就是 myWorker.js 文件，从而实现调用。代码如下所示：

```
var worker = new Worker("myWorker.js");     //定义Worker，并传入参数
```

通常情况下，需要检测一下 Worker 对象是否存在。若不存在，则自动创建一个新的 Worker 对象，然后运行 myWorker.js 中的代码。代码如下所示：

```
if (typeof(worker)=="undefined"){            //未定义，其类型为undefined
   worker=new Worker("myWorker.js ");        //创建一个Worker
}
```

然后就可以从 Web Worker 发送和接收消息。为 Web Worker 对象添加一个 onmessage 事件监听器来接收消息。代码如下所示：

```
worker.onmessage=function(event){    //动态分配事件属性，并绑定无名事件处理函数
   document.getElementById("result").innerHTML=event.data;
                                     //将接收的消息显示在指定的标记内
}
```

event.data 中存放来自新线程 postMessage(data)方法回传的数据 data。

当然 Worker 新线程也可以通过 postMessage(data)方法来向主线程发送数据、绑定 onmessage 方法来接收主线程发送过来的数据。

## 17.4.4 终止 Web Worker

由于 Web Worker 不能自行停止，但是启动它们的页面可以通过调用 terminate()方法停止它们，并释放浏览器/计算机资源。代码如下所示：

```
worker.terminate();      //终止新线程
```

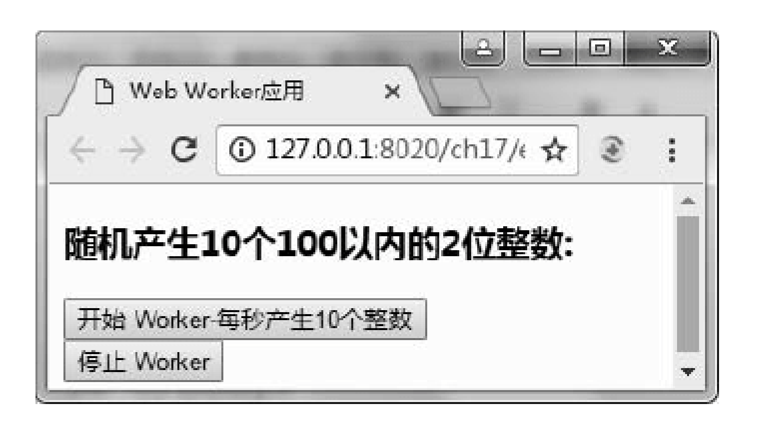
在 Web Worker 应用中还会遇到同时加载多个脚本的情况，此时可以在 Worker 中通过 importScripts(url)加载另外的脚本文件，也可以使用 setTimeout()、clearTimeout()、setInterval()、clearInterval()等方法来定时或周期性地启动或停止相关代码。

**【例 17-4-1】**HTML5 Web Worker 多线程实现每隔 1s 随机产生 1 组 10 个 100 以内的两位整数。代码如下，页面效果如图 17-12 所示。

视频讲解

```
<!-- edu_17_4_1.html -->
<!DOCTYPE html>
<html lang="en">
   <head>
      <title>Web Worker应用</title>
      <meta charset="UTF-8">
   </head>
   <body>
      <h3>随机产生10个100以内的2位整数：</h3>
      <p><output id="result"></output></p>
      <button onclick="startMyWorker()">开始 Worker-每秒产生10个整数
      </button>
      <br/><button onclick="stopMyWorker()">停止 Worker</button>
      <script>
         var worker;                        //定义全局变量
         function $(id){return document.getElementById(id);}
                                            //通过id获取对象
         function startMyWorker(){          //启动我的worker
            if(typeof(Worker)!=="undefined")
                                            //判断浏览器是否支持Web Worker
            {
```

```
                if(typeof(worker)=="undefined")     //判断worker是否存在
                {
                  worker=new Worker("myWorker.js");  //不存在则创建
                }
                worker.onmessage = function (event) {  //捕获传递的消息
                   $("result").innerHTML=event.data; //显示在指定的标记内
                }
            }else{                                   //浏览器不支持Web Worker
             $("result").innerHTML="对不起,您的浏览器不支持Web Worker...";
            }
        }
        function stopMyWorker(){
          worker.terminate();                        //终止线程
        }
    </script>
  </body>
</html>
```




（a）初始状态




（b）开始线程




（c）停止线程

图 17-12　HTML5 工作线程应用

# 17.5　综 合 实 例

利用 IndexedDB 实现简易图书管理系统。页面设计功能要求如下。实现页面内容、页面表现与行为充分分离。HTML 页面采用 HTML5 新增结构标记来设计，主要包括 header、section、nav、footer 等标记，设计三个导航：图书汇总、添加图书、系统设置，采用三个 section 分别设计三个不同的用户界面，页面效果如图 17-13～图 17-19 所示。




图 17-13　初始界面




图 17-14　图书汇总页面

（1）图书汇总页面。“检索图书”按钮功能有两个：不输入任何检索内容时，单击按钮能够实现检索对象仓库中的所有图书。输入检索内容时，单击按钮能够按图书标题检索相关图书。页面效果如图 17-15 和图 17-16 所示。




图 17-15　未输入检索内容检索图书页面

图 17-16　输入 HTML 后的检索图书页面

（2）添加图书页面。“添加图书”按钮功能是将输入的图书标题、作者、ISBN 等信息添加到对象仓库 books 中。页面效果如图 17-17 和图 17-18 所示。




图 17-17　添加图书初始界面




图 17-18　图书添加页面

（3）系统设置页面。系统设置页面中设置三个命令按钮，分别是清除所有图书、清除数据库、数据库初始化。其中“清除所有图书”按钮功能是将 books 对象仓库中的所有对象全部删除。“清除数据库”按钮功能是将整个数据库（myBooks）删除。“数据库初始化”

功能是重新初始化系统，自动添加五种图书信息。页面效果如图 17-19 所示。




图 17-19　系统设置页面

文件命名要求 HTML 文件为 edu_17_5_1.html，CSS 文件为 books.css，JavaScript 文件为 mybooks.js。

- HTML 部分代码。

```
<!-- edu_17_5_1.html -->
<!DOCTYPE html>
<html>
   <head>
      <meta charset="UTF-8">
      <title>IndexedDB管理客户端数据-简易图书管理系统</title>
      <link type="text/css" rel="stylesheet" href="books.css" />
      <script type="text/javascript" src="mybooks.js"></script>
   </head>
   <body calss="list">
      <!--1.采用HTML5设计页面内容  -->
      <header>
         <h2>简易客户端图书管理系统</h2>
         <nav>
            <ul>
               <li><a id="a1" href="#list" class="list">图书汇总</a></li>
               <li><a id="a2" href="#add" class="add">添加图书</a></li>
               <li><a id="a3" href="#setting" class="setting">系统设置</a></li>
            </ul>
         </nav>
      </header>
      <!-- 采用3个section设计3个不同的导航页面  -->
```

```
        <section class="list">
          <!-- 第1 个section是图书列表清单 -->
          <form name="list">
            <input type="search" size="30"  name="query" placeholder="请输入需要查询的图书名称" />
          <input type="button" value="检索图书" onclick="showBooks();"/><br />
          <textarea id="booklist" rows="8" ></textarea> <!-- 显示查询结果 -->
           </form>
        </section>
        <section class="add">
           <!-- 第2 个section是图书添加 -->
           <form name="add">
           <fieldset>
             <legend align="center">图书基本信息</legend>
          图书标题：<input type="text" name="title" autocomplete="true" /><br/>
          作者：<input type="text" name="author" autocomplete="true" required/> <br />
           ISBN：<input type="text" name="isbn" autocomplete="true" pattern="[0-9]{13}" required /><br/>
          <input type="button" class="myButton"  value="添加图书" onclick="addBook();">
           </fieldset>
         </form>
        </section>
        <section class="setting">
           <!-- 第3 个section是系统设置 -->
           <form name="setting">
               <input type="button" value="清除所有图书" class="myButton" onclick="deleteAllBooks();" />
               <input type="button" value="清除数据库"  class="myButton" onclick="deleteDatabase();" />
               <input type="button" value="数据库初始化"  class="myButton" onclick="loadBooks()" />
           </form>
        </section>
        <footer>
           <p>版权所有&copy;2017-2025 HTML5创意与推广工作室</p>
               </footer>
    </body>
</html>
```

- CSS 部分代码。

```
/* books.css */
/* 2.表现设计(定义域和列表框样式) */
form {width: 400px;margin: 5px auto;
    height: 180px;border: 1px double #0066FF;padding: 20px;}
form {text-align: center;}
ul{list-style-type: none;text-align: center;}
ul li {display: inline-block;width: 90px;}
ul li a:link,ul li a:active,
ul li a:visited { text-decoration: none;}
ul li a:hover {border-bottom: 3px solid red;}
h2 {height: 48px;    text-align: center;    background: #EAEAEA;
```

```
   padding-top: 15px;   vertical-align: middle;}
.myButton {margin: 20px auto; width: 100px;
   height: 30px; background: #DADADA;}
/* 初始时所有section均不可见 */
section { display: none;}
/* 当body和section具有相同的class属性时，section则显示 */
body.list section.list,body.add section.add,
body.setting section.setting {display: block;}
footer {padding: 0px; margin: 0 auto;
   text-align: center; background: #DADADA;height: 40px;}
p {padding-top: 10px; font-size: 16px;}
#booklist{width:100%;}
```

- JavaScript 代码。

```
/* mybooks.js */
/* 3.交互行为设计 */
//3.1 系统变量初始化
var db = null; //定义保存数据对象结果集的变量
var request,objStore1;
var DBName = "myBooks",DBVersion = 1; //定义数据库名称和版本号
var bookLists = [{       title: "Web前端开发技术-Html、Css、JavaScript",
       author: "储久良",       isbn: "9787302431695"    },
    {   title: "计算机组成原理(修订版)",        author: "张功萱",
       isbn: "9787302433637"    },
    {   title: "HTML/CSS/JavaScript标准教程",
       author: "本书编委会",       isbn: "9787121079344"    },
    {   title: "HTML+CSS网页设计与布局从入门到精通",
       author: "温谦",       isbn: "9787115183392"    },
    {   title: "Java 2实用教程(第5版)",
       author: "耿祥义",       isbn: "9787302464259 "    }];
//3.2 浏览器的支持判断
var indexedDB = window.indexedDB || window.mozIndexedDB || window.
msIndexedDB || window.webkitIndexedDB;
//3.3 定义 创建indexedDB数据库的方法，并监听3个事件
function createDB(dbName, dbVersion) {
    request = indexedDB.open(dbName, dbVersion);
                                              //返回一个IDBrequest对象
    request.onerror = function(event) {
       alert("打开数据库失败:" + event.target.errorCode);
       console.log("打开数据库失败:" + event.target.errorCode);
    }
    request.onsuccess = function(event) {
       alert("打开数据库成功!");
       //console.log("打开数据库成功!");
       db = event.target.result;                 //给db赋值
       var trans1 = db.transaction(["books"], "readwrite");
       objStore1 = trans1.objectStore("books"); //创建books对象仓库
    }
    request.onupgradeneeded = function(event) {
       alert("版本变化! " +"版本号为" + event.newVersion);
       console.log("版本变化! " + event.newVersion);
       db = event.target.result;
       //为books对象仓库创建事件对象
```

```
        //var trans1 = db.transaction(["books"],"readwrite");
        if(!db.objectStoreNames.contains("books")){//如果不存在，则创建
        objStore1=db.createObjectStore("books", {keyPath: "isbn"});
                                                    //创建对象仓库
        objStore1.createIndex("by_title", 'title', {unique: false});
        objStore1.createIndex("by_author", 'author', {unique: false});
        objStore1.createIndex("by_isbn", 'isbn', {unique: true});
        }
        loadBooks();                                //初始化图书
        window.location.reload();
        window.location.hash = "#list";
        $("booklist").value = "";
    }
}
//3.4 启动创建数据库事件处理程序
createDB(DBName, DBVersion);                        //数据库初始化
function $(id) {
    return document.getElementById(id);
}

function loadBooks() {                              //初始化加载5种图书
    $("booklist").innerHTML = ""; //加载前先清空列表
    alert("开始装载图书....");
    var trans=db.transaction("books","readwrite");
    var objStore1=trans.objectStore("books");
    for(var i = 0; i < bookLists.length; i++) {
                                //采用for循环将3个对象添加到指定对象仓库
        var request = objStore1.put(bookLists[i]);
                                //put存在则更新，不存在则添加对象到books中
        request.onerror = function() {
            console.error('数据库中已有该对象,不能重复添加!! ');
        };
        request.onsuccess = function() { //控制台输出或某个HTML标记内输出
            console.log('对象已成功存入对象仓库中! ')
            //console.dir(request.value);
        };
    }
}
function showBooks() {          //显示所有图书
    var query = document.forms.list.query.value;
    $("booklist").value = "";   //加载前先清空列表
    var transaction = db.transaction(["books"], "readonly");
                                //为books定义只读事件
    var objStore = transaction.objectStore("books");
                                //获取books对象仓库
    var index = objStore.index("by_title"); //按title进行索引
    var range = IDBKeyRange.bound(query,query+"z"); //生成范围range对象
    //alert(range.value);
    var result = (query.length > 0) ? index.openCursor(range) : index.
    openCursor();
    var i = 1;
    result.onsuccess = function(e) {    //打开索引游标，启动成功监听事件
        var cursor = e.target.result;   //获取结果集
        if(cursor) {                    //通过控制台输出相关信息
```

```
            text1 = i + "-" + cursor.value.title + "," +
                          cursor.value.author + "," + cursor.value.isbn;
            $("booklist").value +=text1 + "\n"
            cursor.continue();            //下移游标，迭代执行
            i++;
        } else {
            console.log("没有检索内容");
            //$("booklist").value = "没有检索到所需要的图书！"
        }
    };
    result.onerror = function(e) {console.log("检索失败！")};
}
function addBook() {
    //添加一本图书进入客户端图书库
    var title = document.add.title.value;
    var author = document.add.author.value;
    var isbn = document.add.isbn.value;
    var onebook = {
        title: title,
        author: author,
        isbn: isbn
    };
    if(title.length > 0 && author.length > 0 && isbn.length > 0) {
        var trans2 = db.transaction("books","readwrite");
        var objStore2 = trans2.objectStore("books"); //获取books对象仓库
        var request1 = objStore2.add(onebook);
        trans2.oncomplete = function(event) {
            alert("图书成功添加！界面即将被清空！");
            document.forms.add.title.value = "";
            document.forms.add.author.value = "";
            document.forms.add.isbn.value = "";
            window.location.reload();
            //location.hash="#list";
        };
    }
}
function deleteDatabase() {
    if(indexedDB) {
        var deleteDB = indexedDB.deleteDatabase("books");
        deleteDB.onsuccess = function(e) {
            alert("数据库删除成功，即将重新初始化...");
            window.location.reload();
        }
    }
}
function deleteAllBooks() {
    var trans1 = db.transaction("books", "readwrite");
    var objStore1 = trans1.objectStore("books");
    objStore1.clear();
    trans1.oncomplete = function(e) {
        alert("所有图书清除成功！");
    }
    trans1.onerror = function(e) {
        alert("所有图书清除失败！");
    }
```

```
    window.location.reload();
}
/* 监听hashchange事件，并绑定事件处理函数
      根据hash值，动态地给body修改class属性的值
      完成Section随导航自动切换
 */
window.addEventListener('hashchange', function() {
    switch(location.hash) {
        case "#add":
            document.body.className = "add";
            break;
        case "#setting":
            document.body.className = "setting";
            break;
        default:
            document.body.className = "list";
    }
}, false);
```

使用 Chrome 浏览器查看页面效果，并使用开发者工具（按 F12 键），选择 Application 菜单，从左侧的 Storage 目录中选择 IndexedDB，单击展开对象仓库 books，可以看到索引和具体的存储对象等信息，如图 17-20 所示。




图 17-20　用开发者工具查看 IndexedDB 中的数据

# 本 章 小 结

本章介绍了 HTML5 的一些需要借助于 JavaScript 脚本来完成的功能，主要有客户端存储 Web Storage、画布 Canvas、拖放 Drag & Drop、多线程 Web Worker 等。通过大量的

示范案例讲解了在实际开发中如何运用这些对象的方法和属性。

重点介绍了 Web Storage 和 IndexedDB 等客户端存储技术。其中 localStorage、sessionStorage 对象可以存储少量客户端数据。而 IndexedDB 数据库可以存储大量客户端数据。

HTML5 Canvas 通过 JavaScript 脚本来完成绘图。canvas 标记本身并没有绘图能力，所有的绘制工作必须在 JavaScript 内部完成。可以通过多种方法使用 Canvas 绘制路径、盒、圆、字符以及添加图像。

拖放（Drag 和 Drop）是一种常见的特性，即抓取对象以后拖到另一个位置。任何元素都能够拖放，只要设置 draggable 属性为 true 即可。

Web Workers 允许长时间运行脚本，而不阻塞脚本响应单击或者其他用户交互，它还允许执行长期任务而无须页面保持响应。

# 练习与实验

## 练习 17

**1．选择题**

（1）能够保存 localStorage 对象数据的方法是（　　）。

A．localStorage.setItem(key,value)　　B．localStorage.getItem(key)

C．localStorage.removeItem(key)　　D．localStorage.key(index)

（2）能够从 sessionStorage 对象中读取数据的方法是（　　）。

A．sessionStorage.setItem(key,value)　　B．sessionStorage.key(index)

C．sessionStorage.removeItem(key)　　D．sessionStorage.getItem(key)

（3）sessionStorage.key(index)方法的作用是（　　）。

A．保存数据　　B．读取数据

C．得到某个索引的 key　　D．删除单个数据

（4）HTML5 Canvas 对象的默认宽度为 300px，默认高度是（　　）。

A．200px　　B．300px　　C．400px　　D．500px

（5）Canvas 绘图是借助于 JavaScript 脚本通过（　　）方法进行图像绘制。

A．getElementById()　　B．getContext("2d")

C．fillRect()　　D．strokeRect()

（6）使用 Canvas 绘制圆形的方法是（　　）。

A．beginPath()　　B．arc(x, y, 100, 开始角度,结束角度, 绘制方向)

C．closePath()　　D．A、B、C 三个方法依次执行

（7）绘制直线的方法是（　　）。

A．moveTo(x,y)　　B．lineTo(x,y)　　C．arc()　　D．arcTo()

（8）绘制实心文本正确的方法是（　　）。

A．lineTo(x,y)　　B．moveTo(x,y)

C．strokeText(text,x,y)　　　　D．fillText(text,x,y)

（9）在设置线性渐变时至少需要指定（　　）次颜色停止。

A．2　　B．3　　C．4　　D．1

（10）绘制图像裁剪的方法是（　　）。

A．drawImage(image,x,y)　　　　B．createPattern(image,type)

C．clip()　　　　D．drawImage(image,x,y,width,height)

**2．填空题**

（1）在指定(x,y)处绘制指定宽度为 width、高度为 height 的图像 image 的方法是______________________________；选择从指定位置(sx,sy)开始和指定宽度为 sw、高度为 sy 的区域，在画布上指定的（dx,dy）处绘制宽度为 dw、高度为 dy 的图像 image 的方法是______________________________。

（2）设置放置对象至少必须通过________和________事件属性来监听是否允许拖放和放置。

（3）要使图像 img 标记成为拖放对象，必须设置________属性为 true，同时还要通过________事件属性（事件句柄）来监听 dragstart 事件。

（4）放置对象必须通过________事件属性来监听放置事件，并绑定相关事件处理函数接收被放置的对象。

（5）Web Worker 对象可以通过______运算符来创建。

（6）创建一个新线程（Worker），必须传入一个参数，这个参数是______文件。

（7）在主线程中可以通过给 Worker 对象添加一个________事件属性绑定事件处理函数来接收新线程的数据，新线程可以通过____________方法向主线程传递数据。

（8）线程一旦创建，不能自已停止，可以通过____________方法停止新线程运行，并释放资源。

（9）Web 本地存储常用的两个对象分别是________和________。他们都具有相同的方法，分别是________、________、________、________。

**3．简答题**

（1）两个 Web 本地存储对象在使用中有什么区别？

（2）在 IndexedDB 中，对象仓库的更新与添加方法有什么区别？

（3）简述 Web Worker 的工作原理。主线程和新线程之间是如何进行数据交换的？

## 实验 17

1．运用 HTML5 Web 本地存储设计简易手机通讯录。实现通讯录添加、查询、删除、重置等功能。通讯录内容为姓名、电话。页面效果如图 17-21 所示。

按钮功能设计要求：“添加”按钮的功能是将用户在表单中输入的合法用户姓名和电话添加到 phone 对象仓库中，并同时添加到下面的多行文本域中。“查询”按钮的功能是根据用户在姓名中输入的内容，在对象仓库中查询，并显示在“电话”对应的文本框中。“删除”按钮的功能是根据用户在姓名中输入的内容，查找对应的对象，并删除，同时自动更

新下列多行文本框中的通讯录。“重置”按钮的功能是将表单中输入域中的内容清空。

2．使用 IndexedDB 数据库实现手机通讯录管理功能。效果图如图 17-21 所示。程序设计要求：

（1）创建 phoneInfo 数据库，创建 phone 对象仓库，用于保存每个用户的通讯信息。







图 17-21　简易手机通讯录管理程序页面

（2）手机通讯信息结构为姓名 name、电话 phone。

（3）按钮功能参照实验 17 第 1 题中的要求。

3．上机实验并调试【例 17-3-1】HTML5 拖放图像应用中的全部代码。

4．上机实验并调试【例 17-4-1】中的全部代码。

# 附录 A　模拟试卷 1（120 分）

**一、选择题（每题 1 分，共 30 分）**

1．以下表示 HTML 文档的标记是（　　）。

A．<html></html>　　B．<JavaScript></JavaScript>

C．<style></style>　　D．<body></body>

2．下列（　　）语言可以实现网页交互功能。

A．HTML　　B．CSS　　C．C++　　D．JavaScript

3．以下标记中用于设置页面标题的是（　　）。

A．<html>　　B．<title>　　C．<head>　　D．<caption>

4．下面（　　）是换行符标记。

A．<enter>　　B．<br>　　C．<font>　　D．<p>

5．在 HTML 中，标记<pre>的作用是（　　）。

A．转行标记　　B．标题标记　　C．文字效果标记　D．预排版标记

6．下列不属于字体<font>标记属性的是（　　）。

A．color　　B．face　　C．align　　D．size

7．以下关于列表标记说法错误的是（　　）。

A．<ol>有序列表　　B．<ul>无序列表　C．<dl>定义列表 D．<li>嵌套列表

8．下列选项表示相对路径的是（　　）。

A．images/tu.gif　　B．ftp://219.11.65.123

C．/root　　D．http://www.baidu.com

9．图像名为 myhome.jpg，要访问目标为 http://www.edu.cn，以下创建一个图像链接正确的是（　　）。

A．<a href="http://www.edu.cn" >myhome.jpg</a>

B．<a href="http://www.edu.cn" ><img src="myhome.jpg">

C．<img src="myhome.jpg"><a href="http://www.edu.cn" ></a>

D．<a href="http://www.edu.cn" ><img src="myhome.jpg"></a>

10．在 CSS 文字、排版、边界等的设置上，经常用到长度单位，下列是相对单位的是（　　）。

A．in　　B．pc　　C．cm　　D．px

11．以下关于<select>标记说法正确的是（　　）。

A．<select>定义的表单元素在一个下拉菜单中显示选项

B．rows 和 cols 属性可以定义其大小

C．<select>定义的表单元素是一个单选按钮

D．<select>定义的表单元素通过设置 multiple 属性可以实现多选

12．CSS 文件的扩展名为（　　）。

A．txt　　B．htm　　C．css　　D．html

13．要使表格行高为 18px，以下方法正确的是（　　）。

A．<table border="1" height="18cm" ></table>

B．<tr border="1" height="18pt" ></tr>

C．<tr　height="18px" ></tr>

D．<table border="1" height="18pc" ></table>

14．下列（　　）选项的 CSS 语法是正确的。

A．body:color=black　　B．{body:color=black(body)}

C．body{color:black;}　　D．{body;color:black}

15．边距属性为“上边距：20px、下边距：30px、左边距：40px、右边距：50px”正确的设置是（　　）。

A．margin:20px 30px 40px 50px　　B．border: 20px 30px 40px 50px

C．margin:20px 50px 30px 40px　　D．margin-top:20px 30px 40px 50px

16．在 JavaScript 中，下列满足变量 x 大于等于 20 且小于 100 条件的正确表达式是（　　）。

A．(X>=20 & x<100)　　B．(x>=20 and x<100)

C．(X>=20 or x<=100)　　D．(x>=20 && x<100)

17．设 s1、s2 为字符变量，s1="How Are You!"：s2="a"，则 s1.indexOf(s2)的结果是（　　）。

A．–1　　B．4　　C．5　　D．以上都不是

18．引用外部 compute.js 脚本正确的语法是（　　）。

A．<script href="compute.js">　　B．<style href="compute.js">

C．<script src="compute.js">　　D．<style src="compute.js">

19．下列声明自定义函数 selectNumber()正确的是（　　）。

A．function : selectNumber(){}　　B．function selectNumber(){ }

C．function =selectNumber(){}　　D．function {selectNumber()}

20．下列选项中（　　）是求出两个数中的最大数。

A．Math.ceil(20,50)　　B．Math.max(20,50)

C．Math.min(20,50)　　D．top(20,50)

21．HTML5 的正确 doctype 格式是（　　）。

A．<!DOCTYPE html>

B．<!DOCTYPE HTML5>

C．<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 5.0//EN" "http://www.w3.org/TR/html5/strict.dtd">

D．<!DOCTYPE html5>

22．在 HTML5 中，属于组合标题元素的是（　　）。

A．<group>　　B．<hgroup>　　C．<headings>　　D．<header>

23．用于播放 HTML5 音频文件的正确 HTML5 元素是（　　）。

A．<media>　B．<audio>　C．<video>　D．<movie>

24．在 HTML5 中，为输入字段提供占位符（提示信息）的属性是（　　）。

A．required　B.formvalidate　C．validate　D．placeholder

25．下列输入类型用于定义周和年控件（无时区）的是（　　）。

A．date　B．week　C．year　D．time

26．下列属性中表示 CSS3 的过渡的属性是（　　）。

A．animation　B．transform　C．transition　D．box-shadow

27．下列属性中能够设置圆角边框的属性的是(　　)。

A．box-shadow　B．border-image　C．border-style　D．border-radius

28．下列选项中定义动画 animation 的关键帧的是(　　)。

A．@keyframes　B．keyfrmaes　C．@import url()　D．@keyframe

29．能够保存 localStorage 对象数据的方法是（　　）。

A．localStorage.setItem(key,value)　B．localStorage.getItem(key)

C．localStorage.removeItem(key)　D．localStorage.key(index)

30．使用 Canvas 绘制圆形的方法的是（　　）。

A．beginPath()　B．arc(x, y, 100, 开始角度,结束角度, 绘制方向)

C．closePath()　D．A、B、C 三个方法依次执行

**二、填空题（每空 1 分，共 20 分）**

1．HTML 中标记分为两种：一种__（1）__标记，另一种是__（2）__标记。

2．在 HTML 文件里，版权符号的控制代码是__（3）__。

3．<font>标记常用的属性有 3 个，分别是 color、__(4)__、__（5）__(以首字母排序）。

4．要实现超链接在新窗口中打开目标网页需要将 target 属性值设置为__（6）__。

5．将表格的标题设置为“课程成绩表”的完整 HTML 语句（利用表格标记）是__（7）__。

6．浮动框架一般使用在<__（8）__>标记中，用于在其中打开超链接。

7．定义一个表单元素普通按钮，可通过<input type=__（9）__name="select" id="select" />来定义。

8．CSS 中类选择符用__（10）__来表示，ID 选择符用__（11）__来表示。

9．定义一个变量 today 为日期对象语句是__（12）__。

10．Math.round(Math.random()*100)产生的数据范围是__（13）__。

11．CSS 中样式表的定义有 4 种，分别是行内样式表、__（14）__、__（15）__、__（16）__。

12．插入背景音乐可使用标记__（17）__（不使用 bgsound 标记）。

13．设置表格跨行的属性是__（18）__。

14．加载名为 image1.gif 图像可在<img>标记中这样设置：__（19）__="image1.gif"。

15．使无序列表的列表项符号显示为“○”，可通过<ul type="__（20）__">实现。

**三、看图编程（每空 2 分，共 46 分）**

1．执行下面程序，并在浏览器中打开，效果如图 A-1 所示，根据网页呈现的信息，完善代码。




图 A-1　JavaScript 初步应用

```
<html>
 <head>
       <title>__(1)__</title> <!--写上标题-->
 </head>
 <body>
 <script type="text/javascript">
     __(2)______ //提示信息如图A-1所示
     </script>
   </body>
</html>
```

2．按如图 A-2 所示的页面效果和代码中注释部分的提示信息完善程序代码。




图 A-2　标题字应用

```
<html>
<head>
        <title> 标题字应用 </title>
</head>
<body>
__(3)____<!-- 3号标题 -->
__(4)____    <!—水平线大小：3，颜色：红色，对齐方式：居中  -->
 __(5)____         <!-- 4号标题显示"1.HTML" -->
 __(6)______     <!-- 5号标题显示"2.CSS " -->
       <h6>3.JavaScript </h6>    <!-- 6号标题显示"3.JavaScript " -->
   </body>
</html>
```

3．按如图 A-3 所示的页面效果和代码中注释部分的提示信息完善程序代码。

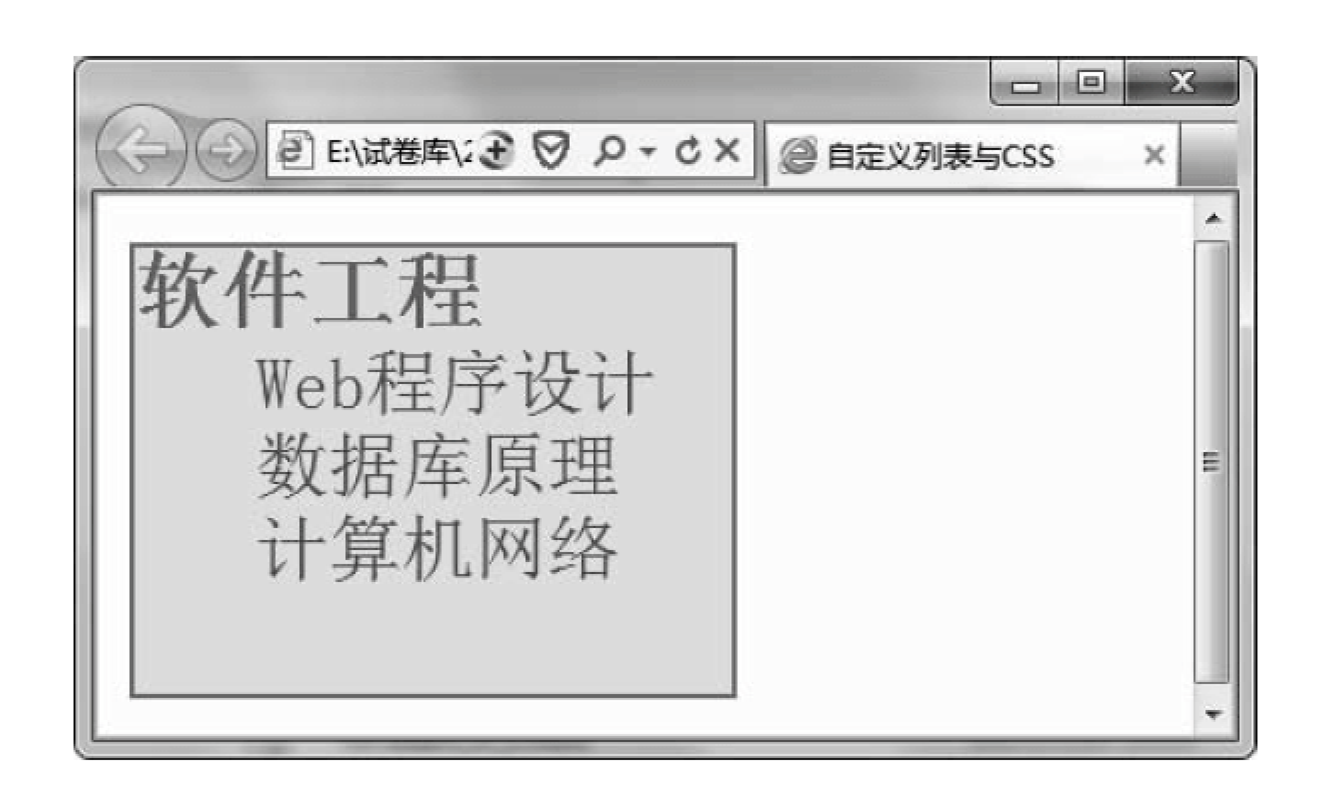



图 A-3　自定义列表与 CSS

```
<html>
 <head>
   <title> 自定义列表与CSS </title>
     <style type="text/css"> /*定义样式表的开始标签 */
     dl{ __(7)__;   /*定义背景颜色为#99ff99*/
         width:200px;height:150px;
         border:__(8)__;}   /*定义边框为2px、双线、颜色为#ff3333*/
     dt{font-size:28px;color:red;font-weight:bold;}
     dd{ color:green;__(9)__;   /*定义字大小为24px*/  }
   </style>
   </head>
  <body>
     __(10)__<!--自定义列表 -->
     __(11)__  <!--定义列表项目为"软件工程"  -->
          <dd>Web程序设计</dd>
          <dd>数据库原理</dd>
          <dd>计算机网络</dd>
   </dl>
 </body>
</html>
```

4．按如图 A-4 所示的页面效果和代码中注释部分的提示信息完善程序代码。

程序功能：单击“投注”按钮随机产生 1 注福利彩票号码，每 1 注号码由 8 个 01～30 之间的整数构成；单击“清空”按钮，将文本框清空。

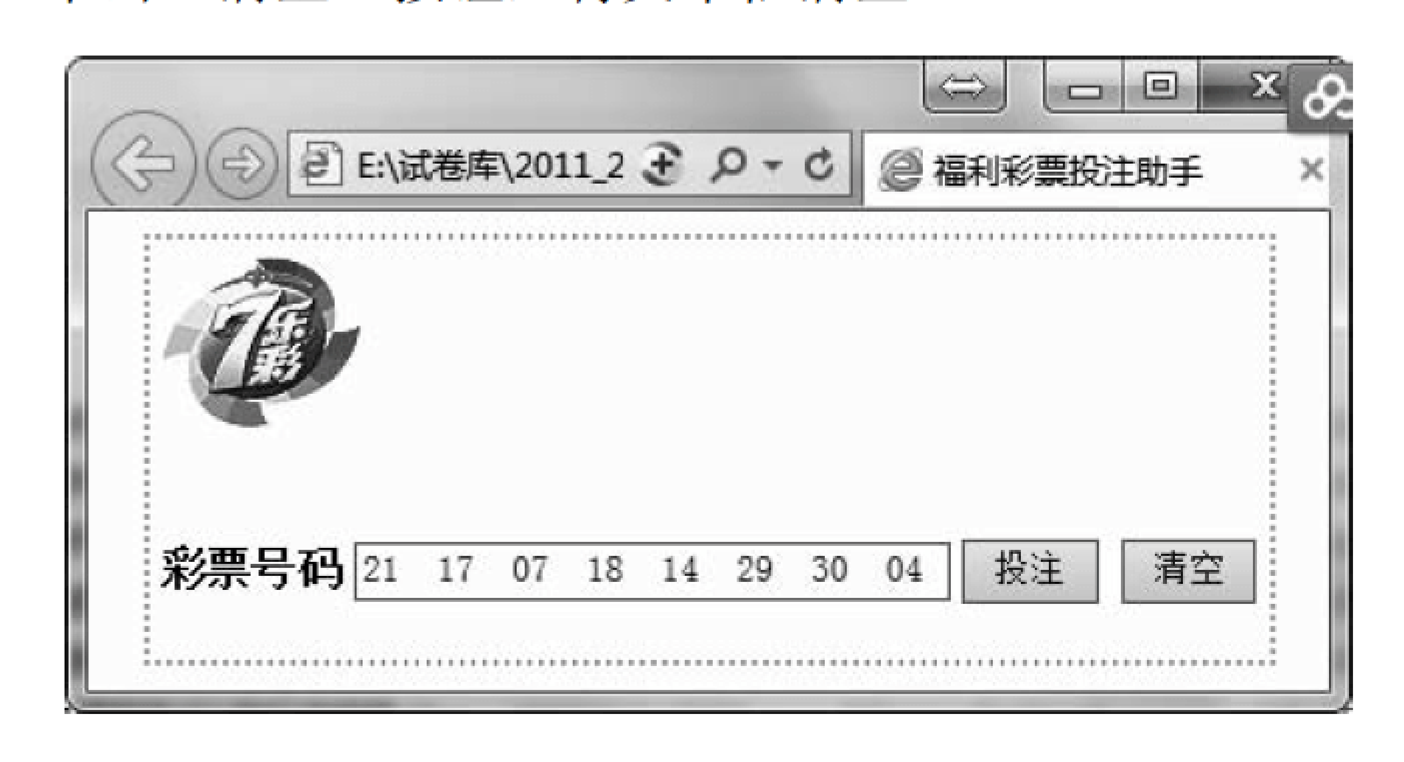



图 A-4　福利彩票投注助手

```
<html>
  <head>
      <title> 福利彩票投注助手 </title>
      <style type="text/css">
        div{background:url("ico_71.gif") left top no-repeat;
         width:450px; height:150px; margin:100px auto; border:2px dotted
         #ff3300;
         __(12)__;  /*图层的文字内容的颜色为白色，值用英文颜色名 */
      }
      form{ margin:0 auto;}
      table{margin:0 auto; __(13)__; /* 13表格的字体样式为粗体 */}
      h2{font-size:24px;text-align:center;}
   </style>
   <script type="text/javascript">
      function selectnumber(num){ //彩票选号助手
       var number=new Array();     //number定义为数组
       var temp=Math.floor(Math.random()*30+1);
       for (i=0;____(14)_____;i++ )//产生8个01～30之间的整数，补充循环判断表达式
      {
        for (var j=0;j<number.length ;j++ )
          {//去除重复号码
            if (temp==number[j])
              {
               temp=Math.floor(Math.random()*30+1);
              j=0;
              }
          }
          number[i]=temp; //下舍入
          if (number[i]<10) { ____(15)_____;}//如果数组元素的值小于10，加上前导"0"
       }
      __(16)_____.value=number.join("  ");//通过ID号获取number1文本框对象，并将
                                          //计算结果赋值给它
   }
  </script>
 </head>
<body>
    <div id="" class="">
        <h2>福利彩票投注助手</h2><br><br>
       <form method="post" action="">
         <table>
           <tr>
            <td>彩票号码</td>
       <!--  17设置number1文本框为只读  -->
      <td><input type="text" name="number1" size="28" id="number1"_(17)_>
      </td>
            <!--  18事件句柄与事件代码绑定-->
            <td><input type="button" value="投注" onclick="__(18)___;">
           <input type="reset" value="清空">
           </td>
          </tr>
          </table>
       </form>
    </div>
 </body>
</html>
```

**5．按如图 A-5 所示的页面效果，完善程序代码（10 分）。**







图 A-5　input 类型与数据列表关联

填充说明：（19）列表属性；（20）设置输入域占位符；（21）数据列表标记；（22）与 input 标记关联的属性；（23）显示带有记号的文本。

```
<!doctype html>
<html lang="en">
 <head>
     <meta charset="UTF-8">
     <title>HTML5页面元素input和datalist标记的应用</title>
</head>
 <body>
    <input ＿(19)＿="courese"  ＿(20)＿="请选择课程" />
    <＿(21)＿ ＿(22)＿="courese">
       <option value="HTML5移动应用开发">
       <option value=".NET应用开发">
       <option value="JavaEE应用开发">
       <option value="PHP+MySQL应用开发">
    </＿(21)＿>
    <p>＿(23)＿注释：＿(23)＿除了IE9和更早版本的IE浏览器以及Safari不支持
    datalist标记，其余均支持。</p>
 </body>
</html>
```

## 四、编程题（12 分）

**1．表格编程（6 分）。**

按照如下要求编程实现表格，如图 A-6 所示。

（1）表格标题为“教材表”；

（2）表格边框宽度为 1、居中对齐、宽度 400px、边框颜色为黑色；

（3）单元格样式为内容水平居中、字体样式为“粗体”，采用内部样式表定义；

（4）表格第 4 行需要进行单元格合并操作。

教材表

| 序号 | 教材名称 | 数量（本） |
|---|---|---|
| 1 | 软件工程 | 64 |
| 2 | 数据结构 | 54 |
| 合计 | | 118 |

图 A-6　教材表

**2．JavaScript 编程（6 分）**

按下面要求编程实现找出 1000 以内所有的平方数（例 $4=2^2$，$9=3^2$， $16=4^2$，……，则 4、9、16 等为平方数），并计算累加和，每行输出 10 个数据。采用 for 循环结构实现计算累加和，效果如图 A-7 所示。

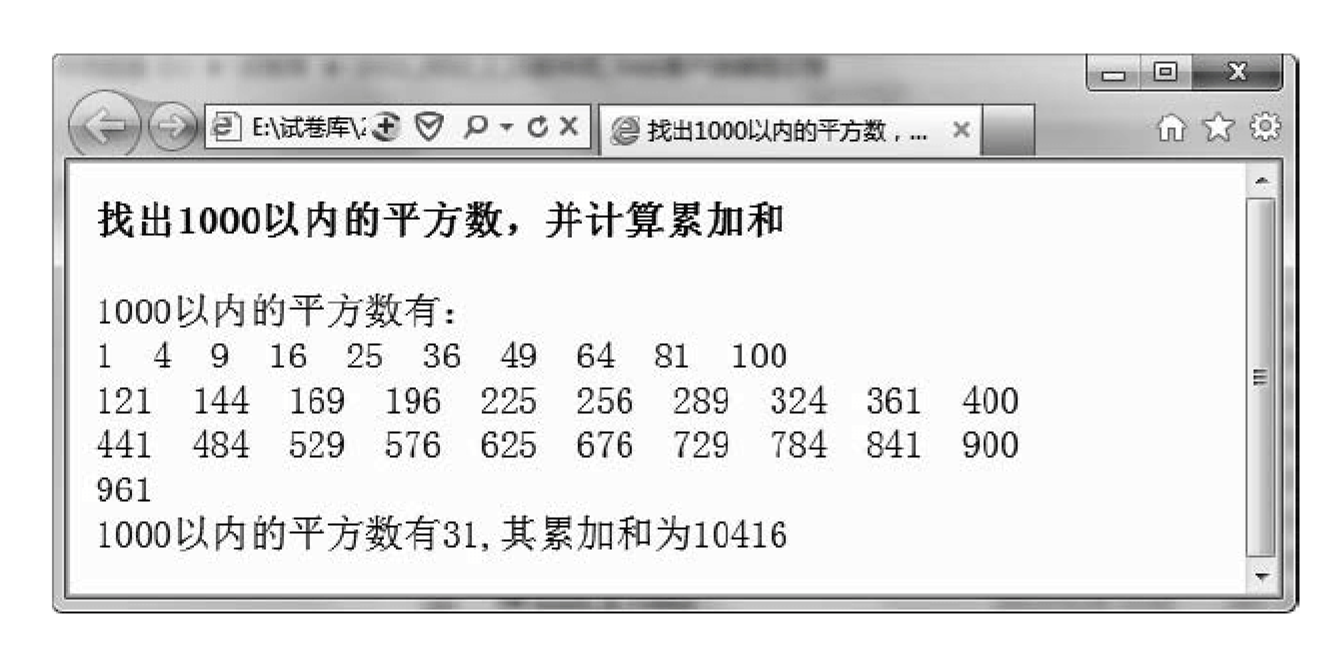



图 A-7　提示信息框界面

**五、问答题（12 分）**

1．举例说明 HTML5 文档的组成结构。（5 分）

2．写出 HTML、CSS、JavaScript 三大部分程序注释的方法。（4 分）

3．JavaScript 关于标识符命名的规定是什么？（3 分）

# 附录 B 模拟试卷 2（120 分）

**一、选择题（每空 1 分，共 30 分）**

1．以下标记中，用于设置页面标题的是（　　）。

A．<html></html>　　B．<head></head>

C．<caption></caption>　　D．<title></title>

2．CSS 文件后缀名通常为（　　）。

A．*.html 或*.htm　　B．*.js　　C．*.css　　D．*.txt

3．以下标记中可以导入外部样式表的标记的是（　　）。

A．<script></script>　　B．<style></style>

C．<link>　　D．<form></form>

4．下列能够实现页面样式与内容相分离的语言是（　　）。

A．CSS　　B．JavaScript　　C．VFP　　D．HTML

5．下列代码中设置 4 号标题字正确的语句是（　　）。

A．<h4>Web 页面设计</h4>　　B．<h*> Web 页面设计</h*>

C．<font size=4>Web 页面设计</font>　　D．<h size=4> Web 页面设计/h4>

6．下列具有字体加粗功能的标记是（　　）。

A．<font></font>　　B．<b></b>

C．<pre></pre>　　D．<center></center>

7．以下关于文本格式标记描述正确的是（　　）。

A．<b>斜体标记　　B．<p>定义列表标记

C．<del>删除线标记　　D．<sub>设置上标

8. 超链接的 target 属性值为（　　）时可以在父窗口中打开目标网页。

A．_self　　B．_blank　　C．_top　　D．_parent

9．使单元格中的内容垂直居中对齐的正确标记是（　　）。

A．<td valign="middle">　　B．<td valign="top">

C．<td align="middle">　　D．<td valign="bottom">

10．以下标记中用于定义表单中文本域的标记是（　　）。

A．<table> </table>　　B．<input type="textarea">

C．<caption></caption>　　D．<textarea></textarea>

11．以下关于<select>标记说法正确的是（　　）。

A．rows 和 cols 属性可以定义其大小

B．<select>定义的表单元素通过设置 multiple 属性可以实现多选

C．<select>定义的表单元素是一个单选按钮
D．单独使用<select>标记就可以生成下拉列表框

12．设置浮动框架名称的属性是（　　）。
A．src　　B．width　　C．height　　D．name

13．在 DOM 中通过对象 id 访问对象的正确方法是（　　）。
A．document.getElementsById("id")
B．document. getElementsByName ("name ")
C．document.getElementById("id")
D．document.getElementsByTagName ("tagname ")

14．通过（　　）属性可以设置字符间距。
A．letter-spacing　　B．text-indent　　C．cellspacing　　D．cellpadding

15．下列 JavaScript 语句中能正确执行的是（　　）。
A．document.printf("Welcome to You!");
B．var x=5;if (x) {alert("Hello World! ");}
C．var z,if;
D．{var x=4,y=9;alert("Hello World! ");

16．在 JavaScript 中，下列表示返回函数运行结果的语句是（　　）。
A．return;　　B．document.write(number);
C．alert(number);　　D．return number ;

17．在 CSS 中设置________属性的值为"none"，可以去除超链接的下画线效果。
A．line-through　　B．text-transform
C．text-decoration　　D．text-indent

18．下列表达式的计算结果为真的是（　　）。
A．(10>–1) && (null==undefined)　　B．false && true
C．(true ||false) &&( ! true)　　D．8=== "8"

19．下面选项表示绝对路径的是（　　）。
A．www.sina.com.cn　　B．ftp://219.153.40.150/software/
C．../a.html　　D．/a.html

20．下列标识符命名合法的是（　　）。
A．switch　　B．1true　　C．if　　D．$mail_123

21．HTML5 之前的 HTML 版本是（　　）。
A．HTML4.9　　B．HTML4　　C．HTML4.01　　D．HTML4.1

22．用于播放 HTML5 视频文件的正确的 HTML5 元素是（　　）。
A．<media>　　B．<audio>　　C．<video>　　D．<movie>

23．在 HTML5 中，规定输入字段是必填的属性是（　　）。
A．required　　B．formvalidate　　C．validate　　D．placeholder

24．下列 HTML5 元素用于显示已知范围内的标量测量的是（　　）。
A．<gauge>　　B．<range>　　C．<measure>　　D．<meter>

25．从 sessionStorage 对象中删除数据的方法是（　　）。

A．sessionStorage.setItem(key,value)　　B．sessionStorage.key(index)

C．sessionStorage.removeItem(key)　　D．sessionStorage.getItem(key)

26．HTML5 Canvas 对象的默认宽度为 300px，默认高度是（　　）。

A．200px　　B．300px　　C．400px　　D．500px

27．Canvas 绘图是借助于 JavaScript 脚本通过（　　）方法进行图像绘制。

A．getElementById()　　B．getContext("2d")

C．fillRect()　　D．strokeRect()

28．绘制圆弧的方法是（　　）。

A．moveto(x,y)　　B．lineTo(x,y)　　C．arc()　　D．arcTo()

29．在设置线性渐变时至少需要指定（　　）次颜色停止。

A．2　　B．3　　C．4　　D．1

30．下列选项中，能够按指定宽度和高度绘制图像的方法是（　　）。

A．drawImage(image,x,y)　　B．createPattern(image,type)

C．clip()　　D．drawImage(image,x,y,width,height)

**二、填空题（每空 1 分，共 20 分）**

1．HTML 文档结构是由__（1）__和__（2）__两部分构成（用标记名）。

2．在 JavaScript 中将赋值语句 sum=sum+1/(2*n-1)转换为复合赋值语句__（3）__。

3．<font>标记常用的属性有 3 个，分别是 color、__（4）__、__（5）__（以首字母为顺序）。

4．有序列表<ol>的 type 属性的取值有__（6）__种。

5．在 HTML 文件中，超链接可以分为内部链接和__（7）__。

6．表格的标题可以使用__（8）__标记来设置，页面标题是使用<title></title>标记来设置。

7．在<form>中设置一个文件选择按钮必须设置<input>标记的 type 属性为"__（9）__"。

8．CSS 规则中的声明部分是由__（10）__和__（11）__两部分构成。

9．要将数组 num 中所有成员用"-"号串接在一起输出，可以使用的方法是 num.__（12）__。

10．在 p{background:#FF00FF;}这个样式中背景颜色使用__（13）__表示方法。

11．CSS 样式优先级从低到高分别是__（14）__、__（15）__、__（16）__、行内样式。

12．字符串 str="JavaScript 易学！"，则 str.indexOf("S")结果是__（17）__。

13．在 CSS 中，定义 ID 样式的开始符是__（18）__list1{list-style-type:none;}。

14．Math.min(100,200,-300)的结果为__（19）__。

15．函数 parseFloat("2014-12-14")的值是__（20）__。

**三、看图编程（每空 2 分，共 46 分）**

1．按如图 B-1 所示的页面效果，完善程序代码（4 分）。

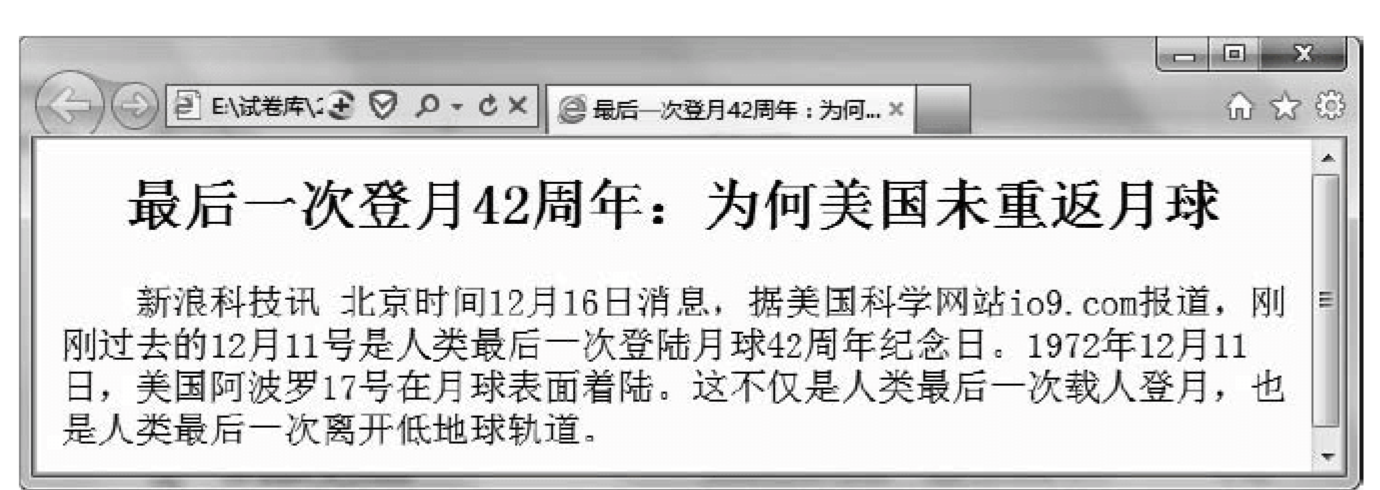



图 B-1　简易 Web 页面

填充说明：（1）设置居中显示；（2）段首空 4 个空格。

```
<html>
 <head>
<title>最后一次登月42周年：为何美国未重返月球</title>
   </head>
   <body>
     <h2 __(1)__> 最后一次登月42周年：为何美国未重返月球</h2>
     <p>__(2)__ 新浪科技讯 北京时间12月16日消息，据美国科学网站io9.com报道，刚刚过
     去的12月11号是人类最后一次登陆月球42周年纪念日。1972年12月11日，美国阿波罗17号
     在月球表面着陆。这不仅是人类最后一次载人登月，也是人类最后一次离开低地球轨道。</p>
   </body>
</html>
```

2．按如图 B-2 所示的页面效果，完成代码填充（8 分）。




图 B-2　超链接与浮动框架应用

填充说明：（3）设置字体大小为 28px；（4）设置边界上下为 0、左右为自动；（5）设置右边的浮动名称为 rigtiframe；（6）设置超链接在右边的浮动框架中打开。

```
<!doctype html>
<html lang="en">
 <head>
  <meta charset="UTF-8">
   <title>浮动框架应用</title>
   <style type="text/css">
    a{width:300px;margin:0 10px;}
    h3{__(3)__;color:#0000ff; text-align:center;}
    div{__(4)__;text-align:center;}
   </style>
</head>
```

```
<body>
   <div id="" class="">
     <h3>浮动框架应用</h3>
     <hr color="red">
   <iframe name="leftiframe" src="http://www.njust.edu.cn" width="300"
   height="300" ></iframe>
     
   <iframe __(5)__ src="https://www.nju.edu.cn/"  width="300" height="300"
   marginwidth="10px"></iframe>
   <p><a href="http://www.gov.cn" target="leftiframe">在左边浮动框架内显示中
   央人民政府网站</a>
  <a href="http://www.moe.gov.cn/"  __(6)__ ="rigtiframe">在右边浮动框架内显
  示教育部网站</a></p>
   </div>
</body>
</html>
```

3．按如图 B-3 所示的页面效果，完成代码填充（12 分）。




2016-2017-1学期成绩表

| 学　号 | 姓　名 | 课程成绩 | | |
|---|---|---|---|---|
| | | 大学英语 | Java程序设计 | Web前端开发技术 |
| 1109520198 | 李婷婷 | 75 | 35 | 76 |
| 1109520199 | 张鹏伟 | 60 | 89 | 66 |

图 B-3　学期成绩表

填充说明：（7）行垂直居中；（8）字体粗细属性；（9）表格边框颜色；（10）行水平居中；（11）跨 3 列居中；（12）应用样式，不及格显示为红色。

```
<html>
 <head>
    <title> 表格应用 </title>
    <style type="text/css">
         tr{text-align:center; vertical-align: __(7)__;height:18px;}
         td{__(8)__:bold}
         .red{color:red;}  /*成绩不及格为红色*/
         </style>
    </head>
   <body>
    <table __(9)__ ="#000011" border="1" width="500px" height="150px">
      <caption><font face="黑体" size="3" color="#000000">2016-2017-1学期成
      绩表</font></caption>
    <tr __(10)__>
        <td rowspan="2">学   号</td>
               <td rowspan="2">姓  名</td>
        <td __(11)__>课程成绩</td>
```

```
        </tr>
        <tr>
         <td>大学英语</td>
             <td>Java程序设计</td>
             <td>Web前端开发技术</td>
         </tr>
       <tr>
                <td>1109520198</td>
                <td>李婷霁</td>
                <td>75</td>
        <td  __(12)__  >35</td>
        <td>76</td>
      </tr>
      <tr>
          <td>1109520199</td>
          <td>张华伟</td>
          <td>60</td>
          <td >89</td>
          <td>66</td>
        </tr>
      </table></body>
    </html>
```

4．按图 B-4 所示的页面效果，完成代码填充（12 分）。

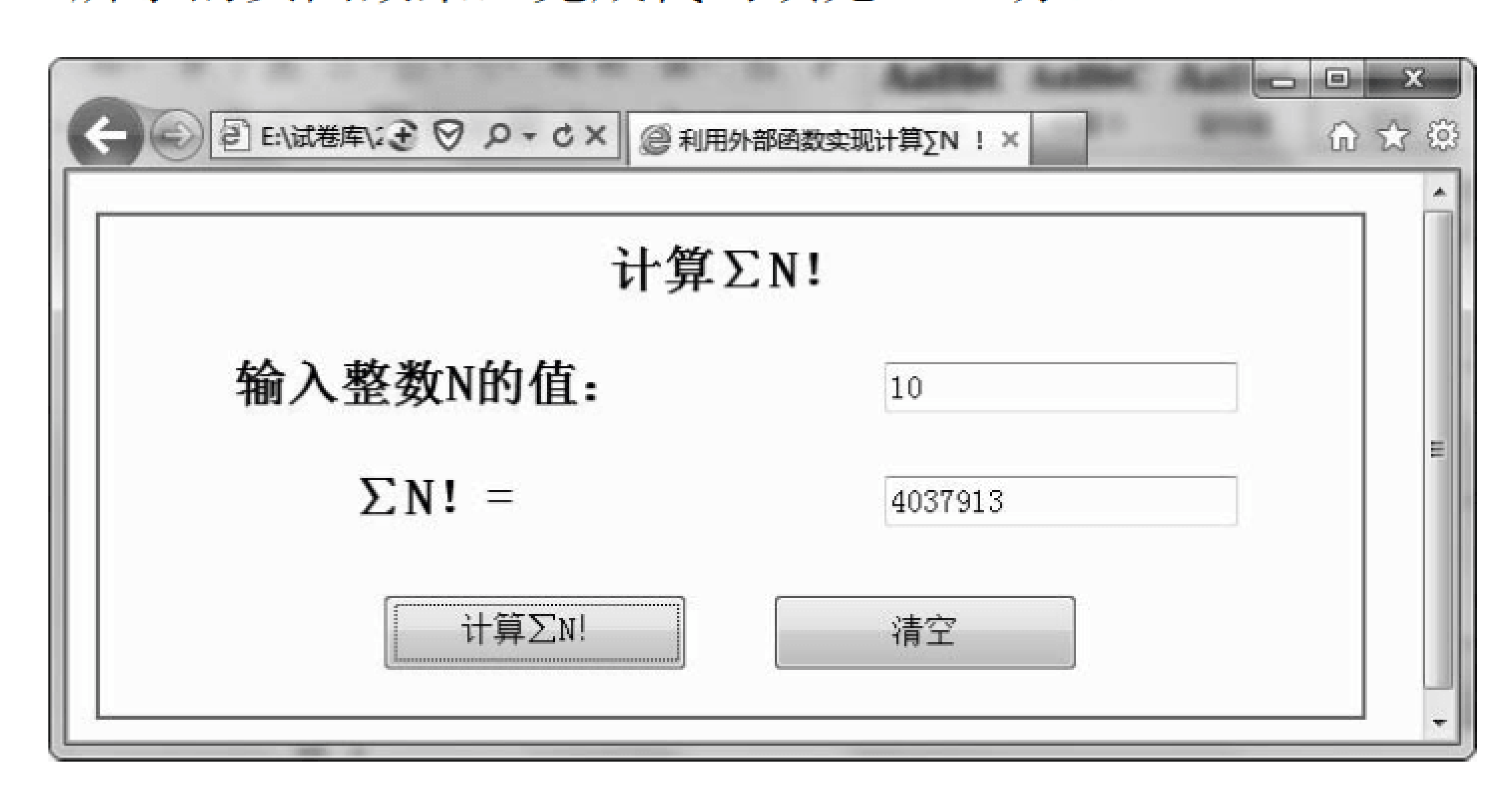



图 B-4　Web 页面

填充说明：（13）调用外部 sum_factorial.js ；（14）累加和文本框只读；（15）定义为按钮；（16）计算累加和赋值语句；（17）调用函数计算累加和；（18）输出累加和。

```
<html>
 <head>
  <title> 利用外部函数实现计算∑N ! </title>
  <script type="text/javascript" __(13)__ ></script>
  <style type="text/css">
    table{width:500px;height:200px; cellspacing:0px;
          margin:0 auto;border:2px solid #339933 ; }
    td{font-size:20px;font-weight:bold;text-align:center;}
    #button{width:120px;height:30px;}
  </style>
</head>
 <body>
    <form method="post" action="">
     <table >
     <tr><td colspan=2>计算∑N! </td></tr>
```

```
     <tr>
           <td>输入整数N的值：</td>
           <td><input type="text" name="" id="n_text"></td>
     </tr>
     <tr><td>∑N! =</td>
          <td><input type="text" name="" id="sum_text" __(14)__ ></td>
    </tr>
    <tr>
        <td colspan="2">
       <input id="button" __(15)___type="button" value="计算∑N!
       " onclick="show();">  
      <input id="button" type="reset" value="  清空 
       ">
       </td>
   </tr>
   </table>
  </form>
 </body>
</html>
// 以下是外部JS文件名为sum_factorial.js
function compute_sum(n){
   var result=1,sum=0;
   for (i=1;i<=n;i++ ) {
            result*=i;
          __(16)___; //累加和}
    return sum;
}
function show(){  //显示累加和的函数
   var n=parseFloat(document.getElementById("n_text").value);
                                                       // 取文本框的值
   var sum= _(17)___;                                  //阶乘计算累加和
        document.getElementById("sum_text").value=__(18)__;//向累加和文本框赋值
   return;                                //结束函数}
```

5．按如图 B-5 所示的页面效果，完善程序代码（10 分）。




图 B-5　教学反馈表

填充说明：(19)（图层 1）放大（1.5 倍）；(20)（图层 2）（X 轴、Y 轴方向）扭曲（30°）；(21) 2D 矩阵转换；(22) 垂直居顶部对齐；((23) 给图层 3 应用#div3 样式。

```
<!doctype html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <title>CSS3 2D转换-扭曲、缩放</title>
  <style type="text/css">
    div{width:100px;height:50px;background:#dadada;border:1px solid
    #00cc66;    }
    #div1{ __(19)__ (1.5,1.5);margin:10px auto;}
    #div2{__(20)__ (30deg,30deg);margin:10px auto;}
    #div3{ __(21)__ (0.866,0.5,-0.5,0.866,20,20);
    /* x轴、y轴缩放0.866;x轴、y轴扭曲0.5和-0.5;x轴、y轴位移20px*/}
    td{text-align:left; __(22)__;}
  </style>
</head>
<body>
    <h3>CSS3 2D转换-缩放、扭曲、矩阵</h3><hr>
    <table border="1px" bordercolor="red" width="750px" height="200px">
        <tr>
            <td>
                <div id="" class=""><p>这是原div</p></div>
                <div id="div1" class=""><p>这个div缩放1.5倍</p></div>
            </td>
            <td>
                <div id="" class=""><p>这是原div</p></div>
                <div id="div2" class=""><p>这个div扭曲方法</p></div>
            </td>
            <td>
                <div id="" class=""><p>这是原div</p></div>
                <div __(23)__ class=""><p>这是div采用matrix方法</p></div>
            </td>
        </tr>
    </table>
</body>
</html>
```

## 四、编程题（12 分）

1．表单编程（6 分）。

编程实现课程教学反馈页面，布局效果如图 B-6 所示。要求如下：

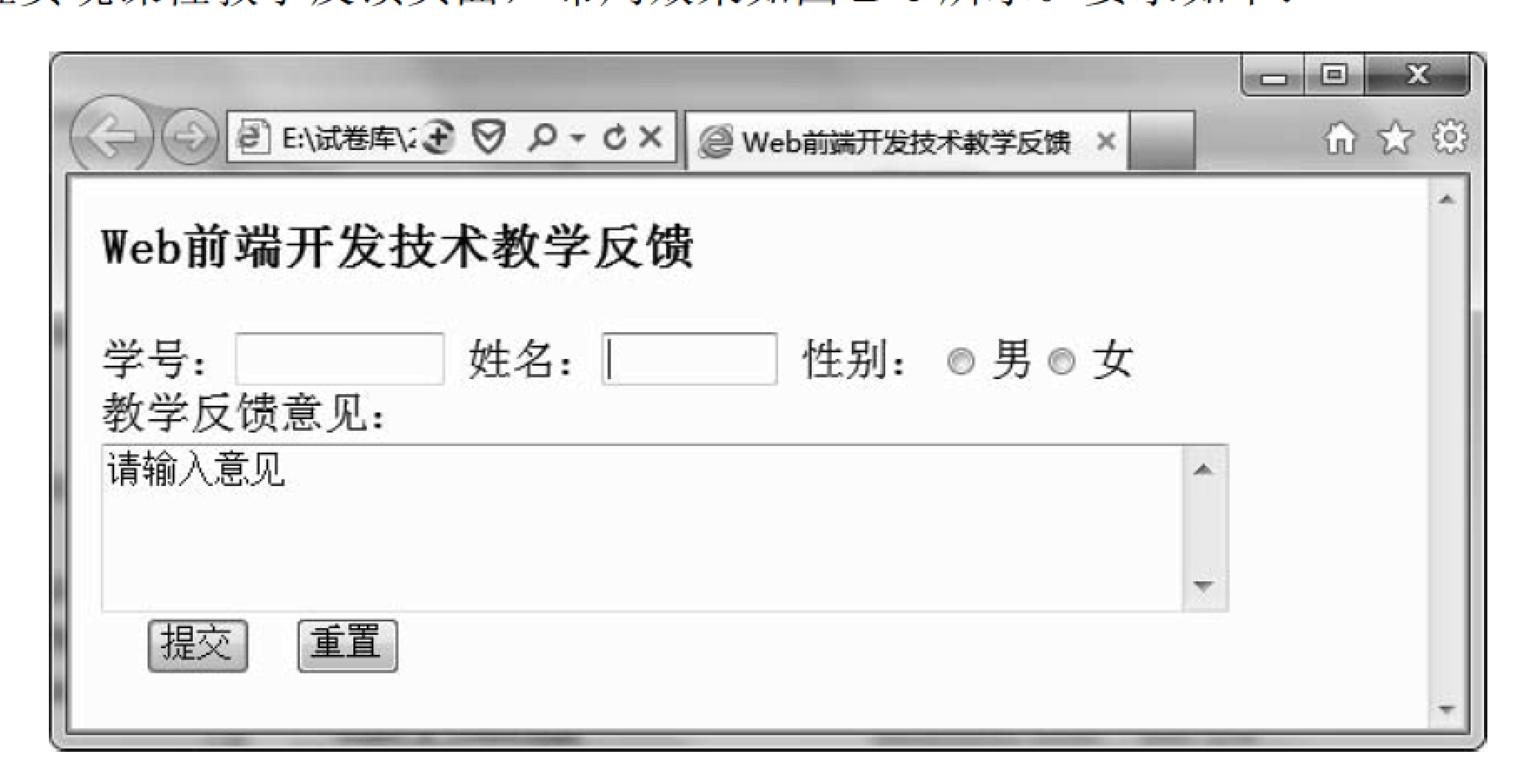



图 B-6　教学反馈表

（1）页面标题为“Web 前端开发技术教学反馈”。

（2）表单中添加 2 个文本框、2 个单选按钮、1 个多行文本区域、1 个提交按钮、1 个重置按钮，其中学号文本框最大长度为 10、姓名文本框最大长度为 8、多行文本区域为 4 行 60 列；性别：两个单选按钮（男、女）。

（3）用 3 号标题设置页面上的“Web 前端开发技术教学反馈”。

2．JavaScript 编程（6 分）

计算所有能被 17 整除的 3 位整数的和，如图 B-7 所示。

（1）编写 sum3()函数，实现计算所有能被 17 整除的 3 位整数的和，要求采用 do while 循环结构进行编程；




图 B-7　计算满足条件的整数和

（2）采用 4 号标题字显示标题；

（3）在循环体内依次输出满足条件的数；

（4）将计算结果直接输出在页面上。

五、问答题（12 分）

1．简述有序列表的定义语法，并说明有序列表的编号有几种，分别是什么。（5 分）

2．举例说明 CSS 中边界 margin 的多种定义方法。（4 分）

3．举例说明 window 对象中常用的消息框函数。（3 分）

# 参 考 文 献


[1] 沈炜，王槐三，等. 感受精彩 Dreamweaver8 经典商业网站大制作. 北京：人民邮电出版社 2007.

[2] 丛书委会. 网页设计与制作实践教程. 北京：清华大学出版社，2005.

[3] 王晓军，田中雨，刘跃军，等. JSP 动态网站开发基础教程与实验指导. 北京：清华大学出版社，2008.

[4] 杨雪雁. 电子商务概论. 北京：北京大学出版社，2010.

[5] 严月浩. 基于. NET 平台的 Web 开发. 北京：北京大学出版社，2011.

[6] 刘涛，编. 网页设计技术. 北京：中国铁道出版社，2006.

[7] 陈尹立，陈国君. 大学计算机应用高级教程. 北京：清华大学出版社，2008.

[8] 本书编委会. HTML、CSS、JavaScript 标准教程实例版. 3 版. 北京：电子工业出版社，2011.

[9] 韩杰. 网页设计与制作. 西安：西安电子科技大学出版社，2010.

[10] 尤克，常敏慧. 网页制作教程. 北京：机械工业出版社，2008.

[11] 潘明寒，刘永华. 网页设计三合一实用教程. 北京：中国电力出版社，2006.

[12] 胡崧，于慧. 网站建设实例大制作. 北京：中国青年出版社，2007.

[13] 黄玉春. CSS+DIV 网页布局技术教程. 北京：清华大学出版社，2012.

[14] 梁胜民，肖新峰，王占中，等. CSS+XHTML+JavaScript 完全学习手册. 北京：清华大学出版社，2008.

[15] 何秀芳，周进，张淑菊. HTML XHTML CSS 网页制作从入门到精通. 北京：人民邮电出版社，2008.

[16] 胡崧. 网页设计技术伴侣 HTML/CSS/JavaScript 范例应用. 北京：中国青年出版社，2006.

[17] 张金霞. HTML 网页设计参考手册. 北京：清华大学出版社，2006.

[18] [美] Elizabeth Castro Bruce Hyslop. HTML5 与 CSS3 基础教程. 7 版. 望以文，译. 北京：人民邮电出版社，2013.

[19] 林珑. HTML5 移动 Web 开发实战详解. 北京：清华大学出版社，2014.

[20] 谢郁. CSS 高效开发实战：CSS3、LESS、SASS、Bootstrap、Foundation. 北京：电子工业出版社，2014.

[21] [英]Rob Crowther，[爱]Joe Lennon，[美]Ash Blue. HTML5 实战. 张怀勇，译. 北京：人民邮电出版社，2015.

[22] 储久良. Web 前端开发技术——HTML、CSS、JavaScript. 2 版. 北京：清华大学出版社，2016.

[23] 储久良. Web 前端开发技术实验与实践——HTML、CSS、JavaScript. 2 版. 北京：清华大学出版社，2016.

[24] (美)马夫罗蒂(Mavrody，S.). HTML5 和 CSS3 快速参考. 姚皓，凌杰，译. 北京：人民邮电出版社，2013.
[25] 孟庆昌，王津涛. HTML5，CSS3，JavaScript 开发手册. 北京：机械工业出版社，2013.
[26] 贾素玲，王强. JavaScript 程序设计. 北京：清华大学出版社，2009.
[27] 黄斯伟，等. HTML 完全使用详解. 北京：人民邮电出版社，2006.
[28] 郭小成. HTML5+CSS3 技术应用完美解析. 北京：中国铁道出版社，2013.